UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

ESCOLA POLITÉCNICA

DEPARTAMENTO DE ELETRÔNICA

MARCHETAWEB – ARTESANATO PERSONALIZADO COM MADEIRA NA REDE

Autor: _____
Leonardo Ribas Cardoso

Orientador: _____
Prof. Antônio Cláudio Gómez de Sousa, M.Sc.

Examinador: _____
Prof. Aloysio de Castro Pinto Pedroza, D.Sc

Examinador: _____
Prof. Felipe M. G. França, Ph.D.

DEL
Julho de 2008

DEDICATÓRIA

DEDICO ESSE TRABALHO AOS MEUS PAIS, QUE NÃO APENAS ME DERAM A VIDA, COMO SEMPRE DERAM A VIDA POR MIM E À MINHA NAMORADA ROBERTA, QUE ME DEU UM NOVO E BELO SENTIDO À VIDA.

AGRADECIMENTOS

AGRADEÇO A TODOS QUE SEMPRE ME AJUDARAM EM TODAS AS MINHAS FASES DA FACULDADE, DESDE MEUS COLEGAS DE TURMA, EM ESPECIAL AOS MEMBROS DO CONSELHO DEL, QUE SEMPRE ME AJUDARAM DURANTE TODOS OS MEUS LONGOS ANOS NA FACULDADE, AOS PROFESSORES DO DEPARTAMENTO DE ENGENHARIA ELETRÔNICA E DE COMPUTAÇÃO, QUE SEMPRE SE MOSTRARAM DEDICADOS E EMPENHADOS EM FAZER COM QUE SEUS ALUNOS ADQUIRISSEM NÃO SÓ CONHECIMENTO TEÓRICO E TÉCNICO, COMO TAMBÉM QUALQUER CONHECIMENTO QUE POSSA SER ÚTIL AOS ALUNOS, AOS MEUS COLEGAS DE ACCENTURE, QUE SEMPRE SE MOSTRARAM COMPREENSÍVEIS DA IMPORTÂNCIA DE UMA BOA FORMAÇÃO NA CAPACITAÇÃO PROFISSIONAL DAS PESSOAS, E NUNCA SE PUSERAM CONTRA NOS MOMENTOS EM QUE EU PRECISEI ME AUSENTAR OU CHEGAR ATRASADO AO TRABALHO POR CONTA DE COMPROMISSOS NA FACULDADE. GOSTARIA DE AGRADECER A TODOS QUE ME APOIARAM, MAS INFELIZMENTE NÃO HÁ COMO CITAR TODOS. MUITO OBRIGADO!

AGRADECIMENTOS MAIS QUE ESPECIAIS:

AO MEU GRANDE AMIGO LEANDRO OURIQUES, POR TODA A AJUDA QUE SEMPRE ME DEU EM TUDO QUE EU PRECISEI, INCLUINDO PRINCIPALMENTE APOIO MORAL E INCENTIVO.

AO MEU ORIENTADOR ANTÔNIO CLÁUDIO, QUE SEMPRE QUE PODE, SE MOSTROU DISPONÍVEL A ME AJUDAR EM TUDO O QUE EU PRECISEI E QUE ME GUIOU NA ELABORAÇÃO DESSE PROJETO.

À MINHA ROBERTA N. FIRMO, PELO APOIO INCONDICIONAL QUE SEMPRE ME DEU E PRINCIPALMENTE POR TER ME DADO UMA NOVA PERSPECTIVA DE VIDA.

AOS MEUS PAIS, PELA DOAÇÃO DE SUAS PRÓPRIAS VIDAS EM PRÓ DA MINHA, FAZENDO COM QUE EU ME TORNASSE O QUE SOU.

RESUMO

CARDOSO, L.R. MARCHETAWEB – ARTESANATO PERSONALIZADO COM MADEIRA NA REDE. ORIENTADOR: ANTÔNIO CLÁUDIO GÓMEZ DE SOUSA. RIO DE JANEIRO: UFRJ/DEL, 2007. PROJETO FINAL DE CURSO.

ESSE TRABALHO DESCREVE A MODELAGEM EM UML E A IMPLEMENTAÇÃO DE UM SISTEMA *WEB* QUE PERMITIRÁ A VENDA DE PRODUTOS ARTESANAIS, FEITOS COM MADEIRA, DE MODO EXCLUSIVO, ONDE O PREÇO DOS PRODUTOS É CALCULADO AUTOMATICAMENTE, DE ACORDO COM AS ESPECIFICAÇÕES INSERIDAS NO SISTEMA PELO CLIENTE. ESSE SISTEMA POSSIBILITA O CADASTRO DE CLIENTES, FORNECEDORES, MATERIAIS E PRODUTOS À VENDA E SUA MODELAGEM FOI FEITA DE MODO A PERMITIR QUE O MESMO SE MODELE A OUTROS OBJETIVOS SIMILARES E QUE SOFRA ALTERAÇÕES FUTURAS PARA ADEQUAÇÃO DO MESMO A NOVAS FUNCIONALIDADES.

PALAVRAS-CHAVE: INFERFACE *WEB*, SISTEMAS DE VENDA, MODELAGEM DE DADOS, ORIENTAÇÃO A OBJETOS, *E-COMMERCE*.

ABREVIATURAS, SIGLAS, SÍMBOLOS E SINAIS

- BANCO DE DADOS: CONJUNTOS DE REGISTROS DISPOSTOS EM ESTRUTURA REGULAR QUE POSSIBILITA A REORGANIZAÇÃO DOS MESMOS E PRODUÇÃO DE INFORMAÇÃO.
- CNPJ: NÚMERO ÚNICO QUE IDENTIFICA UMA PESSOA JURÍDICA JUNTO À RECEITA FEDERAL BRASILEIRA.
- CÓDIGO-FONTE: CONJUNTO DE PALAVRAS ESCRITAS DE FORMA ORDENADA, CONTENDO INSTRUÇÕES EM UMA DAS LINGUAGENS DE PROGRAMAÇÃO EXISTENTES NO MERCADO, DE MANEIRA LÓGICA.
- CPF: REGISTRO DE UM CIDADÃO NA RECEITA FEDERAL BRASILEIRA NO QUAL DEVEM ESTAR TODOS OS CONTRIBUINTES (PESSOAS FÍSICAS BRASILEIRAS OU ESTRANGEIRAS COM NEGÓCIOS NO BRASIL).
- *E-COMMERCE*: TIPO DE TRANSAÇÃO COMERCIAL FEITA ESPECIALMENTE ATRAVÉS DE UM EQUIPAMENTO ELETRÔNICO, COMO, POR EXEMPLO, UM COMPUTADOR.
- EDR: MODELO DIAGRAMÁTICO QUE DESCREVE O MODELO DE DADOS DE UM SISTEMA COM ALTO NÍVEL DE ABSTRAÇÃO.
- HARDWARE: PARTE FÍSICA DO COMPUTADOR, OU SEJA, É O CONJUNTO DE COMPONENTES ELETRÔNICOS, CIRCUITOS INTEGRADOS E PLACAS, QUE SE COMUNICAM ATRAVÉS DE BARRAMENTOS.
- HTML: PROTOCOLO DE COMUNICAÇÃO (NA CAMADA DE APLICAÇÃO SEGUNDO O MODELO OSI) UTILIZADO PARA TRANSFERIR DADOS POR INTRANETS E PELA *WORLD WIDE WEB*.
- INTERNET: CONGLOMERADO DE REDES EM ESCALA MUNDIAL DE MILHÕES DE COMPUTADORES INTERLIGADOS PELO PROTOCOLO DE INTERNET QUE PERMITE O ACESSO A INFORMAÇÕES E TODO TIPO DE TRANSFERÊNCIA DE DADOS.
- INTERPRETADORES: PROGRAMAS DE COMPUTADOR QUE LÊEM UM CÓDIGO FONTE DE UMA LINGUAGEM DE PROGRAMAÇÃO E OS CONVERTEM EM CÓDIGO EXECUTÁVEL.
- JAVASCRIPT: LINGUAGEM DE PROGRAMAÇÃO CRIADA PELA NETSCAPE EM 1995.
- LOGIN: CONJUNTO DE CARACTERES SOLICITADO PARA OS USUÁRIOS QUE POR ALGUM MOTIVO NECESSITAM ACESSAR ALGUM SISTEMA COMPUTACIONAL.

- NAVEGADOR: PROGRAMA QUE HABILITA SEUS USUÁRIOS A INTERAGIREM COM DOCUMENTOS VIRTUAIS, CHAMADOS DE HTML (*LINGUAGEM DE HIPERTEXTO*), HOSPEDADOS EM UM SERVIDOR *WEB*, DE ACESSO À INTERNET.
- PF / PESSOA FÍSICA: SER HUMANO PERCEBIDO ATRAVÉS DOS SENTIDOS E SUJEITO AS LEIS FÍSICAS.
- PJ / PESSOA JURÍDICA: ORGANIZAÇÃO QUE A LEI TRATA, PARA ALGUNS PROPÓSITOS, COMO SE FOSSE UMA PESSOA DISTINTA DE SEUS MEMBROS, RESPONSÁVEIS OU DONOS.
- *SCRIPT*: SÉRIE DE INSTRUÇÕES FORMAIS ESCRITAS PARA UM INTERPRETADOR.
- SENHA: PALAVRA OU UMA AÇÃO SECRETA PREVIAMENTE CONVENCIONADA ENTRE DUAS PARTES COMO FORMA DE RECONHECIMENTO.
- SERVIDOR *WEB* / PROVEDOR DE SÍTIOS: PROGRAMA DE COMPUTADOR RESPONSÁVEL POR ACEITAR PEDIDOS HTTP DE CLIENTES, GERALMENTE OS NAVEGADORES, E SERVI-LOS COM RESPOSTAS HTTP, INCLUINDO OPCIONALMENTE DADOS, QUE GERALMENTE SÃO PÁGINAS *WEB*, TAIS COMO DOCUMENTOS HTML COM OBJETOS EMBUTIDOS (IMAGENS, ETC.);
- SGBD: CONJUNTO DE PROGRAMAS DE COMPUTADOR (*SOFTWARES*) RESPONSÁVEIS PELO GERENCIAMENTO DE UMA BASE DE DADOS.
- SÍTIO: CONJUNTO DE PÁGINAS *WEB*, ISTO É, DE HIPERTEXTOS ACESSÍVEIS GERALMENTE PELO PROTOCOLO HTTP NA INTERNET.
- UML: LINGUAGEM DE MODELAGEM NÃO PROPRIETÁRIA DE TERCEIRA GERAÇÃO.
- *WEB*: SISTEMA DE DOCUMENTOS EM HIPERMÍDIA QUE SÃO INTERLIGADOS E EXECUTADOS NA INTERNET.

ÍNDICE DO TEXTO

ÍNDICE DE FIGURAS

-

1. INTRODUÇÃO

1.1. História da Marchetaria

Marchetaria, ou marqueteria, é a arte ou técnica de ornamentar as superfícies planas de móveis, painéis, pisos, tetos, através da aplicação de materiais diversos, tais como: madeira, metais, madrepérola, pedras, plásticos, marfim e chifres de animais, tendo como principal suporte a madeira. Por milênios a marchetaria foi basicamente isso, incrustações de pedras e marfim, que em relação ao Egito antigo só evoluiriam para os motivos geométricos. O mais antigo objeto embutido é uma bacia de pedra calcária da Mesopotâmia, datado por volta de 3000 a.C.. A primeira técnica utilizada, *tarsia certosina*, consistia no recorte de elementos do material a ser utilizado (pedra, madeira, metal etc.) e a posterior incrustação nas cavidades abertas nas superfícies maciças com o auxílio de formões ou ferramentas similares. Para sua fixação utilizava-se cola. No mundo antigo, não há de se desprezar também a importante colaboração romana com o *intarsio*, uma montagem a partir de madeiras de diferentes tonalidades que depois de séculos esquecido retomou seu brilho no século XIV e teve em Florença se principal centro de produção .

Após um tímido, mas fundamental desenvolvimento na Alemanha, Inglaterra e Países Baixos durante os séculos XVI e XVII, a marchetaria se estabelece no país que a revolucionaria: a França. Em meados do século XVII, pelas mãos de Jean Macé, formado pelos mestres de Middelburg, a marchetaria vira moda nesse país. Retomando uma técnica italiana chamada de *Tarsia incastro*, André Charles Boulle, o maior nome da marchetaria e ebenesteria, inventa seu famoso método de corte em superposição.

Com o passar do tempo, a Marchetaria entra em uma fase de decadência, mantida por não mais do que uma centena de artistas, ressurgindo porém com o advento da Art Nouveau; aparecem então os motivos estilizados tais como: flores, pássaros, borboletas, insetos etc.

Atualmente existem na Europa, América do Norte e Austrália muitos ateliês de marchetaria e associações de marcheteiros dispostos a não somente manter as antigas tradições da Arte em Madeira com refinadas criações artísticas de caráter contemporâneo, mas também a restauração de obras antigas.

Hoje, a marchetaria é uma arte rara, mas apesar disso, ou talvez justamente por isso, é uma das mais impressionantes das artes decorativas. De acordo com a técnica utilizada pode-se construir objetos tridimensionais, esculturas, utilitários, jóias etc.

As principais técnicas utilizadas atualmente são:

- *Tarsia a toppo* ou *marquetery a bloc* - Marchetaria maciça, utilizada na fabricação de utilitários, bijuteria, filetes decorativos, esculturas.
- *Tarsia* geométrica - Recorte de motivos geométricos para revestimento de móveis, lambris, caixas, painéis internos, mesas, cadeiras.
- *Marqueterie de paille* - Marchetaria de palha (folhas de plantas desidratadas). Mesmas aplicações da *tarsia* geométrica.
- *Tarsia a incastro* ou *technique* Boulle - Recorte simultâneo das partes a serem montadas.
- *Procéde classique* ou *element par element* - Recorte separado das partes a serem montadas.

1.2. Descrição do Sistema

O sistema desenvolvido trata de um comércio eletrônico (*e-commerce*), onde a transação comercial se dá entre o vendedor de produtos artesanais feitos com madeira, usando a técnica da marchetaria e seus clientes e/ou fornecedores de matérias-primas.

O diferencial do sistema fica por conta da possibilidade que o cliente possui de personalizar seus produtos e saber por meio desse, qual será o novo valor do produto.

Inicialmente, o visitante do sítio tem a possibilidade de visualizar um mostruário dos produtos disponíveis, seus respectivos preços e simular as alterações de preços sofridas pelos produtos após esses terem sido personalizados. Essa visualização é possível para qualquer usuário que acesse o sítio, uma vez que para isso não é necessária a criação de nenhum tipo de cadastro.

Para que haja a possibilidade de comprar quaisquer produtos, é necessário que o usuário efetue um cadastro no sistema, no qual informará dentre outras coisas, se é um cliente pessoa física (PF) ou pessoa jurídica (PJ).

Aos clientes que desejarem personalizar seus produtos, lhes serão apresentadas algumas opções de personalização, como por exemplo: as dimensões do produto, se o mesmo possui ou não alças, ou se será revestido internamente com veludo, ou se possuirá algum desenho específico feito com marchetaria, dentre outras. De acordo com essas

personalizações, o valor final do produto poderá sofrer alterações que o sistema irá informar ao usuário à medida que o mesmo for selecionando suas opções.

Após o cliente escolher o produto desejado, ele efetuará um pedido de compra. Essa solicitação passará pelo vendedor que poderá aceitá-la ou não. Caso seja aceita, o vendedor cadastra no sistema a data prevista para a entrega do pedido e inicia o processo de produção. No momento de entrega do pedido, o vendedor cadastra a data na qual a entrega foi feita, bem como observações julgadas pertinentes ao processo de entrega.

O cliente pode observar a qualquer instante qual o estado dos seus pedidos acessando uma tela de consulta dos seus pedidos.

O sistema possui na sua base de dados, além dos dados referentes aos cadastros dos clientes e dos fornecedores, cadastros dos produtos, materiais e personalizações possíveis. Esses três últimos cadastros são inseridos pelos vendedores. No entanto, a responsabilidade pela manutenção dos tipos de produtos, materiais e personalizações na base de dados, é dada ao administrador do sistema, que também administra os cadastros de todos os usuários do sistema.

1.3. Sumário

Nesse primeiro capítulo, é mostrada a motivação do sistema e um resumo sobre o que o sistema se propõe a realizar. O capítulo seguinte explicita a metodologia utilizada no desenvolvimento do sistema, mostrando-se como o sistema foi criado e quais medidas foram tomadas no seu desenvolvimento. O terceiro capítulo remete à modelagem UML desenvolvida na fase de análise do sistema, enquanto que no quarto capítulo a modelagem UML é referente à fase do projeto do sistema. A seguir, no capítulo cinco, mostra-se a modelagem do banco de dados do sistema e todas as tabelas criadas para o sistema. No sexto capítulo são mostradas características e funcionalidades dos *softwares* utilizados para o desenvolvimento do sistema, desde sua análise até a fase posterior à produção do código-fonte. Por fim, no sétimo capítulo, há a conclusão do trabalho e informações sobre trabalhos futuros e no oitavo capítulo são citadas as referências utilizadas, sejam elas livros ou sítios da internet.

2. METODOLOGIA

Enquanto os requisitos do sistema estavam sendo definidos, os *softwares* que serviriam como suporte ao seu desenvolvimento foram configurados em uma máquina. Estes *softwares* eram o servidor *web* Apache 2.2.8, o interpretador PHP 5.2.5 e o servidor de banco de dados PostgreSQL 8.2.4.1. Uma vez preparado o ambiente de desenvolvimento, a análise do sistema continuou sendo feita através do diagrama de Casos de Uso, do diagrama de classe e do diagrama de comunicação. Além disso, o banco de dados também começou a ser modelado.

Quando as sucessivas versões dos modelos do banco de dados já não apresentavam mudanças significativas, decidiu-se criar uma base de dados inicial para o sistema começar a ser desenvolvido. Como a versão da linguagem PHP utilizada no desenvolvimento do sistema já suportava orientação a objetos, o sistema foi desenvolvido utilizando-se uma classe para cada entidade do modelo do banco de dados, essas classes não possuem atributos, somente métodos, que representam as operações que a classe realiza diretamente com o banco de dados, ou seja, estas classes praticamente possuem os métodos de inclusão, exclusão, alteração e consulta nas suas equivalentes tabelas do banco de dados. Esta solução adotada contribui para a divisão do *software* em camadas e o motivo dessa contribuição é explicada nos próximos parágrafos.

Um *software*, geralmente, pode ser divido em três camadas para sua melhor compreensão e manutenção: Apresentação, Controle e Dados. No caso deste projeto, a camada Apresentação funde-se, em parte, com a camada de Controle; esta, por sua vez, se funde também em parte com a camada de Dados. A fusão entre a camada Apresentação com a camada Controle pode ser explicada porque usamos a linguagem PHP para gerar o código HTML dinamicamente. Além disso, o código PHP é embutido no código HTML estático. Já a fusão entre a camada Controle e a camada Dados é representada pelas classes cujos métodos realizam as interações com o banco de dados.

Na camada Controle também foram utilizados outros dois recursos que contribuem para modularizar o sistema. Um recurso foi a criação de uma classe que faz a conexão do sistema com o servidor PostgreSQL, além disso ela contém métodos que são utilizados pelo *script* de criação do banco de dados desenvolvido em PHP.

O outro recurso foi a utilização de *javascripts* para validar os dados preenchidos nos formulários e fornecer mais interatividade e dinamismo ao sistema. Caso o usuário não informe os dados obrigatórios de um formulário, o sistema exibe um alerta indicando que há informações que não foram preenchidas, desta maneira, o *javascript* é acionado e evita que os dados sejam enviados e com isso não há necessidade de carregar uma outra página que indicasse a ocorrência do erro. Esta forma de validação é mais rápida, reduz o tráfego cliente-servidor uma vez que é executada no cliente, e garante uma maior integridade das informações que são enviadas ao gerenciador do banco de dados. Alguns desses *scripts* também ocultam e exibem trecho do código HTML, em tempo de execução, ou seja, após as páginas já terem sido carregadas no navegador, o que não seria possível implantar usando somente linguagem HTML.

O Apache, o PHP e o PostgreSQL estão sendo utilizados porque a gratuidade das ferramentas era um requisito do projeto, já que a tendência mundial é utilizar *softwares* livres. Essas ferramentas, que funcionam muito bem ao trabalharem em conjunto, estão sempre evoluindo e suas atualizações podem ser obtidas nos seus respectivos sítios oficiais.

Numa última etapa, o sistema foi implantado em um provedor de sítios e está em fase de testes para entrar de fato em produção. Muito embora no desenvolvimento do sistema o mesmo tenha sido constantemente testado, no servidor de produção, ele vem apresentando alguns erros que até o momento em que esse texto foi escrito estão sendo estudados para que sejam feitas as correções.

3. MODELAGEM COM UML – ANÁLISE

3.1. Requisitos de Usuário

1. Requisitos de Alto Nível

REQUISITO	*DESCRIÇÃO*
Req01	Permitir aos clientes personalizar seus produtos.
Req02	Permitir a visualização do valor do produto personalizado aos clientes.
Req03	Permitir a visualização de um mostruário de produtos com seus respectivos preços para qualquer visitante.
Req04	Permitir o cadastro de clientes do tipo pessoa física no sistema.
Req05	Permitir o cadastro de clientes do tipo pessoa jurídica no sistema.
Req06	Permitir a classificação dos clientes de acordo com as aprovações de seus pedidos.
Req07	Permitir o cadastro dos fornecedores de matérias primas.
Req08	Permitir a consulta do vendedor, às matérias primas cadastradas no sistema.
Req09	Permitir que o sistema impeça dois usuários de usarem os mesmos CPF ou CNPJ.
Req10	Permitir ao vendedor a possibilidade de associar possibilidades de personalizações aos produtos.
Req11	Permitir ao vendedor a possibilidade de associar materiais aos produtos.
Req12	Permitir ao cliente visualizar as possibilidades de personalização dos produtos.
Req13	Permitir que o administrador cadastre administradores, vendedores, fornecedores e clientes.
Req14	Permitir ao cliente, efetuar um pedido de compra.
Req15	Permitir que o vendedor aceite ou não um pedido de compra.
Req16	Permitir que o cliente acompanhe o andamento do seu pedido de compra.
Req17	Permitir que os vendedores cadastrem produtos.
Req18	Permitir que os vendedores cadastrem materiais.
Req19	Permitir que os vendedores cadastrem personalizações.
Req20	Permitir que os usuários mantenham seus dados cadastrais atualizados no sistema.
Req21	Permitir ao administrador a manutenção dos tipos de produtos e de materiais.
Req22	Permitir ao vendedor informar ao sistema sobre a entrega dos produtos.
Req23	Permitir que os usuários cadastrados realizem o login no sistema.
Req24	Permitir que o administrador desative administradores, vendedores, clientes e fornecedores.

2. *Requisitos Não-Funcionais*

REQUISITO	*DESCRIÇÃO*
ReqNF01	O sistema será desenvolvido em ambiente *web* e possuirá controle de acessos;
ReqNF02	O sistema utilizará o SGBD PostgreSQL 8.2.4.1;
ReqNF03	O sistema será disponibilizado através de um servidor *web* Apache 2.2.8;
ReqNF04	O sistema será desenvolvido na linguagem PHP 5.2.5;
ReqNF05	O usuário precisará possuir um navegador *web* com acesso a internet para acessar o sistema;
ReqNF06	A resolução ideal para a visualização do sistema será 1024 x 768;
ReqNF07	O navegador *web* deverá estar habilitado a trabalhar com frames.
ReqNF08	Toda documentação e códigos-fonte serão fornecidos ao cliente.

3. *Regras de Negócio*

REGRA	*DESCRIÇÃO*
RN01	Usuário é definido como sendo qualquer pessoa que acesse o sistema. São considerados usuários *visitantes*, *administradores*, *vendedores* e *clientes*.
RN02	O usuário que possui o perfil de *vendedor* tem permissão para consultar e alterar todos os pedidos de compra, enquanto que o usuário que possui o perfil de *cliente* tem permissão apenas para consultar seus próprios pedidos.
RN03	O *visitante* é o usuário que não possui login no sistema.
RN04	O login e a senha dos usuários cadastrados no sistema serão de responsabilidade dos usuários.
RN05	O *cliente*, o *vendedor* e o *administrador* precisam possuir um login cadastrado no sistema.
RN06	O preço do material é considerado como sendo o preço do último fornecimento.
RN07	O preço do produto é aquele que foi cadastrado por último.
RN08	Os logins são individuais, e cada usuário deverá possuir um.
RN09	Os CPF são individuais, e cada usuário do tipo PF deverá possuir um.
RN10	Os CNPJ são individuais, e cada usuário do tipo PJ deverá possuir um.
RN11	A alteração gerada pelas personalizações dos produtos sobre os preços dos mesmos se dá por variações dos valores percentuais dos preços básicos dos produtos supracitados.

3.2. Atores

Os atores, nos diagramas de Casos de Uso, representam os papéis desempenhados pelos diversos usuários que poderão utilizar de alguma maneira os serviços e funções do sistema.

Os atores do sistema desenvolvido são:

1. Administrador

É o responsável pela manutenção das informações de apoio ao sistema e dos cadastros de todos os usuários do sistema.

2. Cliente

É o usuário que utiliza o sistema para conhecer e adquirir os produtos disponibilizados à venda.

3. Fornecedor

É o usuário responsável pela venda das matérias-primas necessárias à fabricação dos produtos postos à venda.

4. Vendedor

É o responsável pelo cadastro e manutenção dos cadastros dos produtos, matérias-primas e personalizações disponíveis no sistema, bem como pela avaliação dos fornecedores e dos pedidos de compras gerados pelos clientes.

5. Visitante

É o usuário que acessa o sistema sem cadastro e cuja funcionalidade no sistema se restringe a conhecer os produtos e as possibilidades de personalizações dos mesmos.

3.3. Diagramas UML

Os diagramas UML objetivam fornecer múltiplas visões do sistema a ser modelado, analisando-o e modelando-o sob diversos aspectos, procurando-se assim atingir a completitude da modelagem, permitindo que cada diagrama complemente os outros.

1. Diagramas de Casos de Uso

O Diagrama de Casos de Uso é o diagrama mais geral e informal da UML e é normalmente utilizado nas fases de Levantamento e Análise de Requisitos do sistema, embora venha a ser consultado durante todo o processo de modelagem e possa servir de base para outros diagramas.

Os casos de uso podem ser aplicados para captar o comportamento pretendido do sistema que está sendo desenvolvido, sem ser necessário especificar como esse comportamento é implantado.

Nesse sistema, houve a necessidade da criação de apenas um Diagrama de Casos de Uso, apresentado na Ilustração 1, que por motivos de visibilidade foi dividido em outros cinco diagramas, um para cada ator.

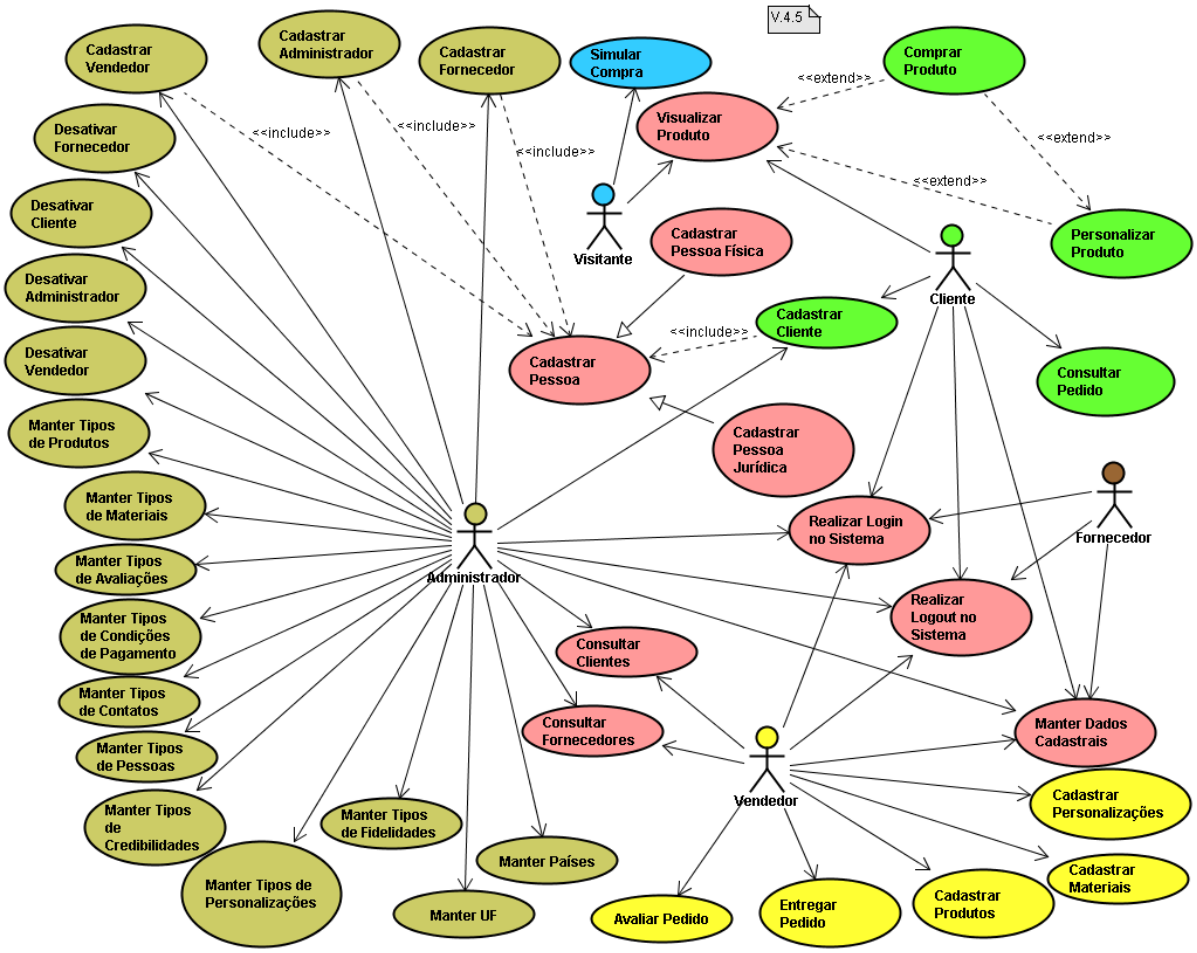

Ilustração 1 - Diagrama de Casos de Uso - Completo

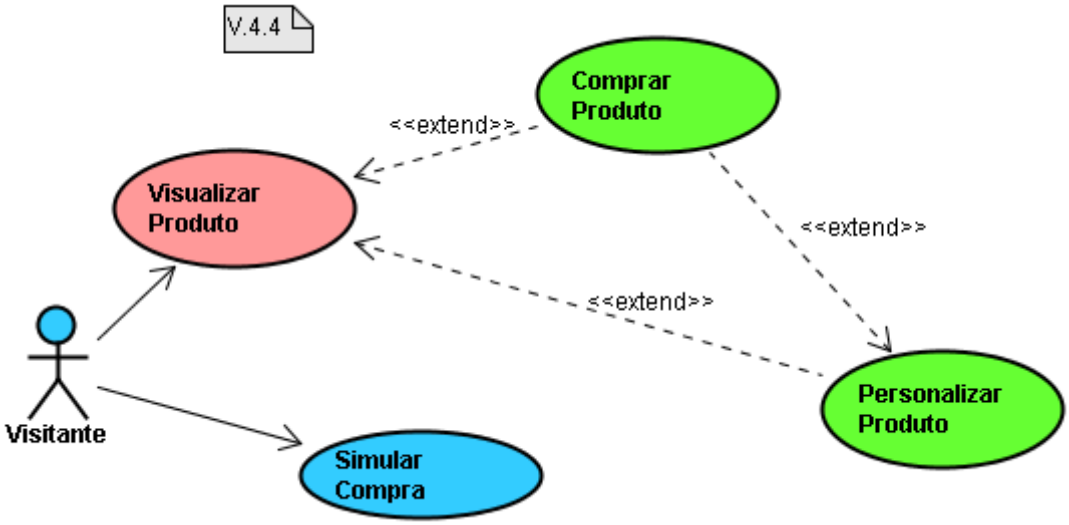

Ilustração 2 - Diagrama de Casos de Uso – Visitante

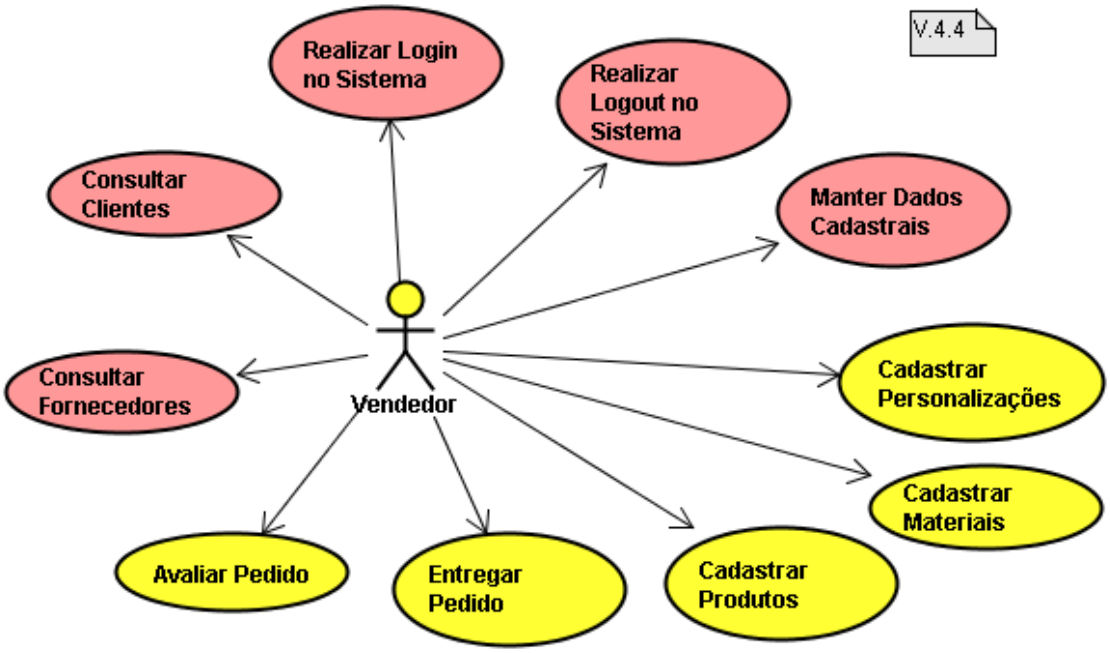

Ilustração 3 - Diagrama de Casos de Uso – Vendedor

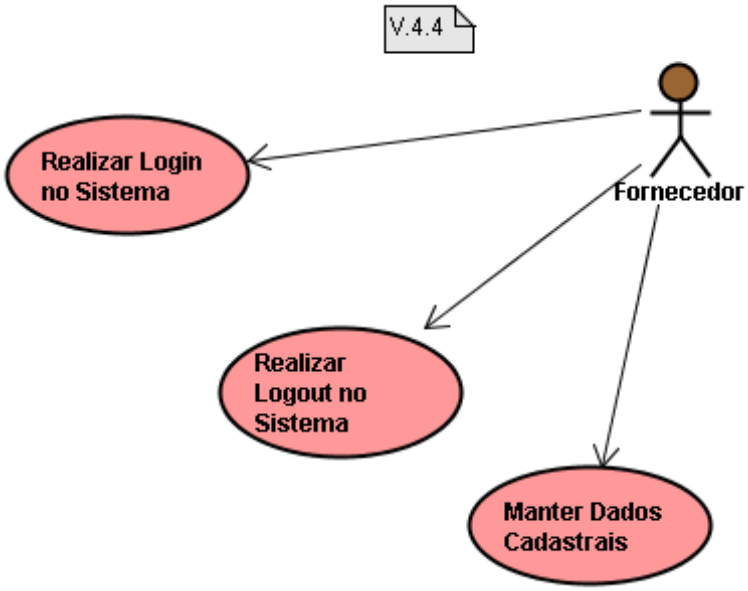

Ilustração 4 - Diagrama de Casos de Uso – Fornecedor

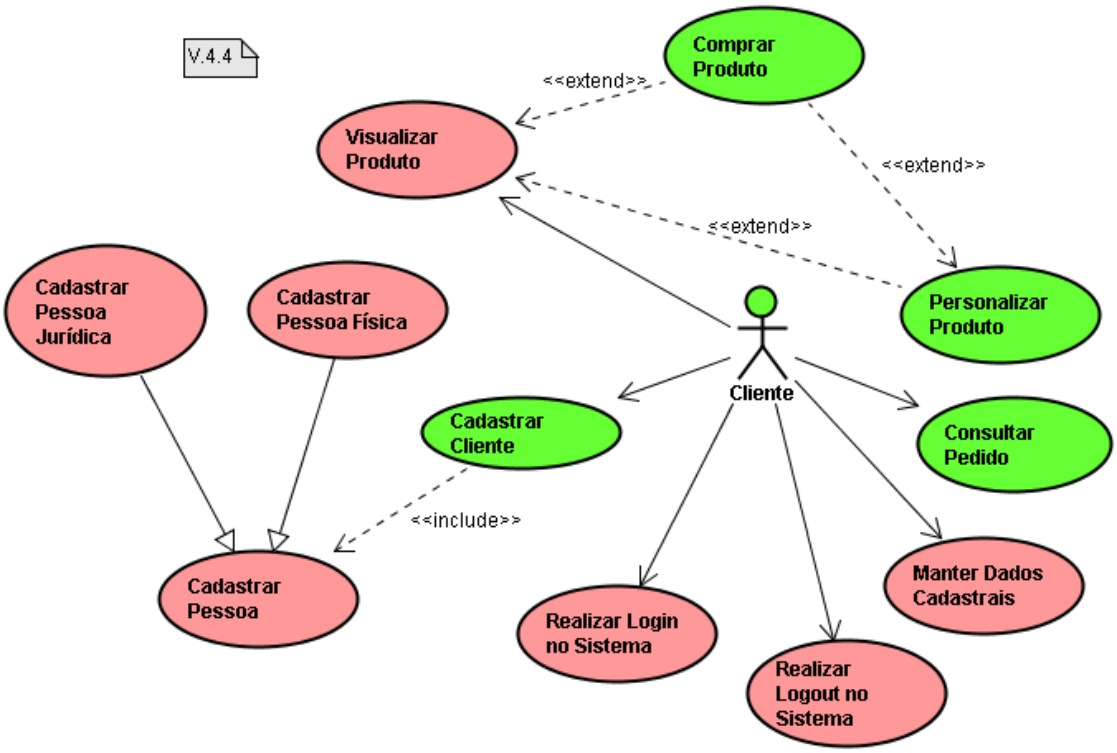

Ilustração 5 - Diagrama de Casos de Uso – Cliente

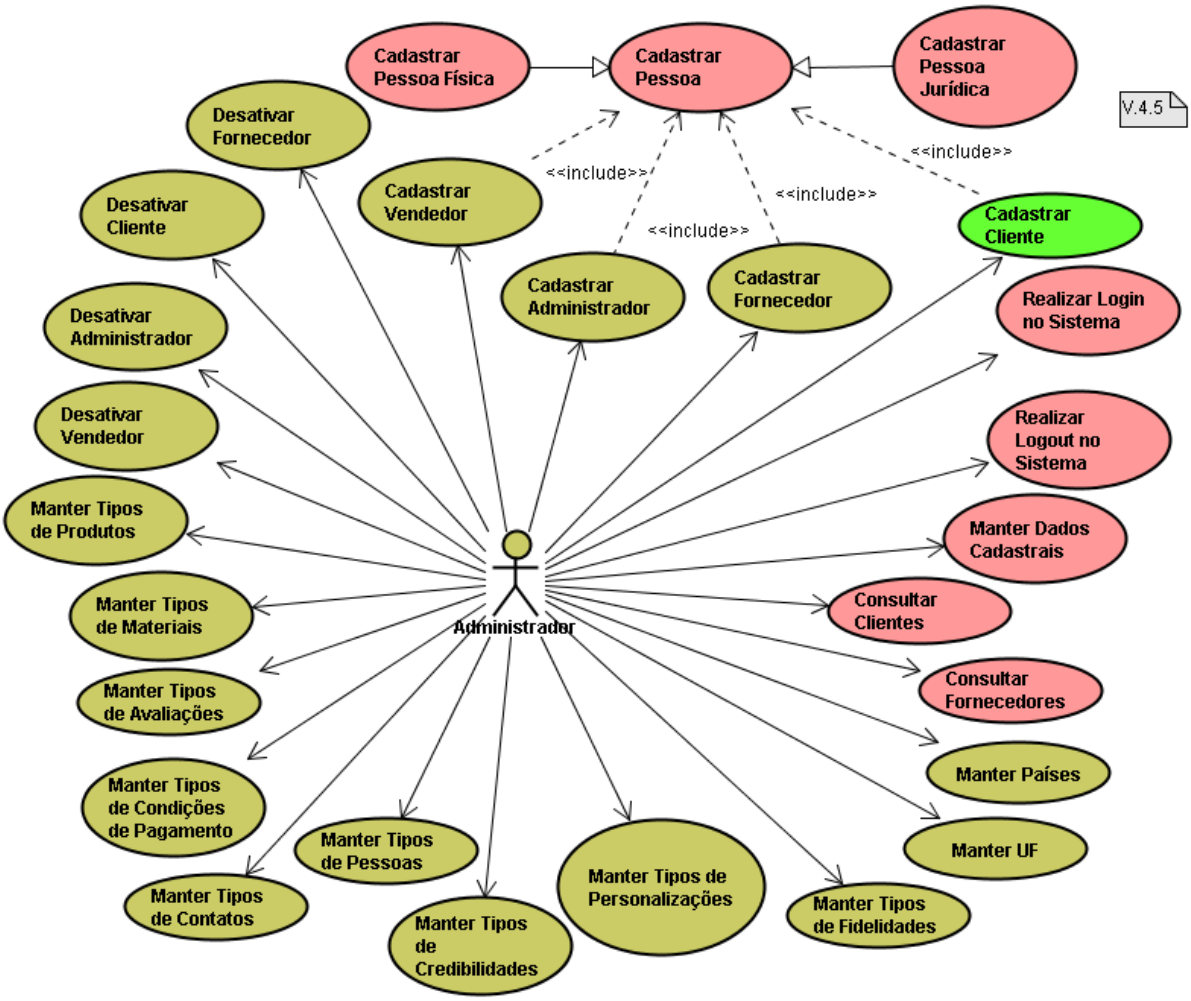

Ilustração 6 - Diagrama de Casos de Uso – Administrador

3.3.1.1. Caso de Uso Nº01: Visualizar Produto

Nome do caso de uso	Visualizar Produto	
Descrição	O ator deseja visualizar os produtos disponíveis no sistema.	
Atores	01. Visitante 02. Cliente	
Pré-Condições	N/A	
Requisitos de Alto Nível	Req03.	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Solicitar visualizar produtos.		
		02. Permitir a visualização dos produtos.
Pós-Condições	N/A	

3.3.1.2. Caso de Uso Nº02: Personalizar Produto

Nome do caso de uso	Personalizar Produto
Descrição	O ator deseja personalizar um produto.

Atores	01. Visitante 02. Cliente	
Pré-Condições	Casos de uso nº01 e nº14.	
Requisitos de Alto Nível	Req01, Req02, Req12.	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Escolher o produto que será personalizado.		
		02. Exibir o produto que será personalizado.
		03. Exibir as possibilidades de personalização do produto.
04. Escolher as personalizações do produto.		
		05. Exibir, à medida que o produto é personalizado, as alterações no seu preço.
		06. Apresentar ao ator a possibilidade de escolher a quantidade de produtos desejados.
07. Escolher a quantidade de produtos desejados.		
		08. Atualizar o preço final, de acordo com as alterações na quantidade de produtos.
Pós-Condições	N/A	

3.3.1.3. Caso de Uso Nº03: Comprar Produto

Nome do caso de uso	Comprar Produto	
Descrição	O ator deseja comprar um produto.	
Atores	Cliente	
Pré-Condições	Casos de uso nº01 e nº09.	
Requisitos de Alto Nível	Req14	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Solicitar a compra dos produtos.		
		02. Gerar um pedido de compra.
Pós-Condições	N/A	

3.3.1.4. Caso de Uso Nº04: Consultar Pedido

Nome do caso de uso	Consultar Pedido	
Descrição	O ator deseja consultar sobre os estados dos seus pedidos de compra.	
Atores	01. Cliente 02. Vendedor	
Pré-Condições	Caso de uso nº03.	
Requisitos de Alto Nível	Req14	
Fluxo Básico		
Ações do Ator 01	Ações do Ator 02	Ações do Sistema
		01. Disponibilizar ao Ator 02 as informações sobre os pedidos de compra.

	02. Atualizar os estados dos pedidos de compra.	
03. Solicitar informações sobre os seus pedidos de compra.		
		04. Exibir os estados atuais dos pedidos de compra.
Pós-Condições	N/A	

3.3.1.5. Caso de Uso Nº05: Cadastrar Pessoa

Nome do caso de uso	Cadastrar Pessoa
Descrição	O ator deseja incluir um cadastro no sistema.
Atores	01. Administrador 02. Cliente
Pré-Condições	Casos de uso nº08, nº20, nº21 e nº22.
Requisitos de Alto Nível	Req04, Req05, Req07 e Req13.
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de cadastro de pessoas.	
	02. Solicitar a informação sobre o tipo de cadastro: PF ou PJ.
03. Escolher o tipo de pessoa que será cadastrado.	
Pós-Condições	Casos de uso 06 e 07.

3.3.1.6. Caso de Uso Nº06: Cadastrar Pessoa Física

Nome do caso de uso	Cadastrar Pessoa Física
Descrição	O ator deseja incluir um cadastro de pessoa física no sistema.
Atores	01. Administrador 02. Cliente
Pré-Condições	Caso de uso nº05.
Requisitos de Alto Nível	Req04 e Req09.
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Escolher o cadastro de PF.	
	02. Solicitar informações para o cadastro de pessoa física.
03. Preencher o cadastro com os dados corretos.	
	04. Validar os dados do cadastro. *(EX1)*
	05. Verificar se o login solicitado já existe na base de dados do sistema. *(EX2)*
	06. Verificar se o CPF solicitado já existe na base de dados do sistema. *(EX3)*
	07. Cadastrar os dados do ator na base de dados.
Fluxo de Exceção – EX1 – Cadastro preenchido com dados incorretos.	
Ações do Ator	Ações do Sistema
	01. Informar o erro e solicitar um novo preenchimento do cadastro.

02. Preencher o cadastro com os dados corretos. *(FB03)*	
Fluxo de Exceção – EX2 – Login previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o login já existe na base e cancelar o atual cadastro.
02. Criar um login inexistente na base do sistema. *(FB03)*	
Fluxo de Exceção – EX3 – CPF previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o CPF já existe na base e cancelar o atual cadastro.
02. Cadastrar um CPF inexistente na base do sistema. *(FB03)*	
Pós-Condições	N/A

3.3.1.7. Caso de Uso Nº07: Cadastrar Pessoa Jurídica

Nome do caso de uso	Cadastrar Pessoa Jurídica
Descrição	O ator deseja incluir um cadastro de pessoa jurídica no sistema.
Atores	01. Administrador 02. Cliente
Pré-Condições	Caso de uso nº05.
Requisitos de Alto Nível	Req05 e Req09.
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Escolher o cadastro de PJ.	
	02. Solicitar informações para o cadastro de pessoa jurídica.
03. Preencher o cadastro com os dados corretos.	
	04. Validar os dados do cadastro. *(EX1)*
	05. Verificar se o login solicitado já existe na base de dados do sistema. *(EX2)*
	06. Verificar se o CNPJ solicitado já existe na base de dados do sistema. *(EX3)*
	07. Cadastrar os dados do ator na base de dados.
Fluxo de Exceção – EX1 – Cadastro preenchido com dados incorretos.	
Ações do Ator	Ações do Sistema
	01. Informar o erro e solicitar um novo preenchimento do cadastro.
02. Preencher o cadastro com os dados corretos. *(FB03)*	
Fluxo de Exceção – EX2 – Login previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o login já existe na base e cancelar o atual cadastro.
02. Criar um login inexistente na base do sistema. *(FB03)*	
Fluxo de Exceção – EX3 – CNPJ previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o CNPJ já existe na base e

	cancelar o atual cadastro.
02. Cadastrar um CNPJ inexistente na base do sistema. *(FB03)*	
Pós-Condições	N/A

3.3.1.8. Caso de Uso Nº08: Cadastrar Cliente

Nome do caso de uso	Cadastrar Cliente	
Descrição	O ator deseja realizar o cadastro de um cliente no sistema.	
Atores	01. Administrador 02. Cliente	
Pré-Condições	N/A	
Requisitos de Alto Nível	Req04, Req05 e Req13.	
Fluxo Básico		
Ações do Ator 01	Ações do Ator 02	Ações do Sistema
01.1a. Acessar a tela de cadastro de pessoas.		
01.2a. Solicitar o cadastro de um cliente.	01.1b. Acessar a tela de cadastro de clientes.	
		02. Solicitar as informações necessárias para o cadastro do cliente.
Pós-Condições	Caso de uso nº05	

3.3.1.9. Caso de Uso Nº09: Realizar Login no Sistema

Nome do caso de uso	Realizar Login no Sistema
Descrição	O ator deseja efetuar seu acesso personalizado ao sistema.
Atores	01. Cliente 02. Fornecedor 03. Vendedor 04. Administrador
Pré-Condições	Caso de uso nº05.
Requisitos de Alto Nível	Req23
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de login e senha.	
	02. Solicitar o login e a senha do ator.
03. Preencher os campos específicos com seu login e senha.	
	04. Validar o login e a senha informados com os dados cadastrados no sistema. *(EX1)*
	05. Permitir acesso à área específica ao tipo de ator que efetuou o login.
Fluxo de Exceção – EX1 – Login ou senha incorretos ou inexistentes.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o login ou a senha estão incorretos.
02. Preencher corretamente o login e a senha. *(FB03)*	
Pós-Condições	N/A

3.3.1.10. Caso de Uso Nº10: Realizar Logout no Sistema

Nome do caso de uso	Realizar Logout no Sistema	
Descrição	O ator deseja desfazer seu acesso personalizado ao sistema.	
Atores	01. Cliente 02. Fornecedor 03. Vendedor 04. Administrador	
Pré-Condições	Caso de uso nº09.	
Requisitos de Alto Nível	N/A	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Solicitar sair do sistema.		
		02. Realizar a saída do ator do sistema.
Pós-Condições	N/A	

3.3.1.11. Caso de Uso Nº11: Manter Dados Cadastrais

Nome do caso de uso	Manter Dados Cadastrais
Descrição	O ator deseja manter seus dados cadastrais no sistema.
Atores	01. Cliente 02. Fornecedor 03. Vendedor 04. Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	Req20
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos dados cadastrais.	
	02. Apresentar a tela de manutenção dos dados cadastrais.
03. Selecionar entre a consulta dos dados cadastrais e a alteração dos mesmos. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Consulta dos dados cadastrais.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os dados cadastrais ao ator.
(b)Fluxo Alternativo – AL2 – Alteração dos dados cadastrais.	
Ações do Ator	Ações do Sistema
	04b. Apresentar os dados cadastrais ao ator.
05b. Alterar os dados cadastrais.	
	06b. Validar os dados recém cadastrados. *(EX1b)*
	07b. Armazenar os novos dados cadastrais no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar os erros. (AL05b)
02. Corrigir os erros.	
Pós-Condições	N/A

3.3.1.12. Caso de Uso Nº12: Consultar Clientes

Nome do caso de uso	Consultar Clientes	
Descrição	O ator deseja consultar os clientes cadastrados no sistema.	
Atores	01. Administrador 02. Vendedor	
Pré-Condições	Caso de uso nº08.	
Requisitos de Alto Nível	Req06	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Solicitar o acesso à tela de consulta dos clientes.		
		02. Apresentar ao ator a tela de consulta dos clientes com as informações sobre as fidelidades dos mesmos.
Pós-Condições	N/A	

3.3.1.13. Caso de Uso Nº13: Consultar Fornecedores

Nome do caso de uso	Consultar Fornecedores	
Descrição	O ator deseja consultar os fornecedores cadastrados no sistema.	
Atores	01. Administrador 02. Vendedor	
Pré-Condições	Caso de uso nº22.	
Requisitos de Alto Nível	N/A	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Solicitar o acesso à tela de consulta dos fornecedores.		
		02. Apresentar ao ator a tela de consulta dos fornecedores com suas respectivas credibilidades. *(AL1)*
Fluxo Alternativo – AL1 – Alterar a credibilidade do fornecedor.		
Ações do Ator		Ações do Sistema
03. Solicitar a alteração da credibilidade do fornecedor.		
		04. Permitir a alteração da credibilidade do fornecedor.
05. Alterar a credibilidade do fornecedor.		
		06. Atualizar no sistema a nova credibilidade do fornecedor. *(FB02)*
Pós-Condições	N/A	

3.3.1.14. Caso de Uso Nº14: Cadastrar Personalizações

Nome do caso de uso	Cadastrar Personalizações
Descrição	O ator deseja cadastrar personalizações no sistema.
Atores	Vendedor
Pré-Condições	Casos de uso nº09 e nº33.
Requisitos de Alto Nível	Req19
Fluxo Básico	

Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao cadastro de personalizações de produtos.	
	02. Solicitar ao ator o cadastro das personalizações dos produtos.
03. Cadastrar as personalizações possíveis para os produtos.	
	04. Armazenar no sistema a nova personalização de produtos.
Pós-Condições	N/A

3.3.1.15. Caso de Uso Nº15: Cadastrar Materiais

Nome do caso de uso	Cadastrar Materiais
Descrição	O ator deseja cadastrar matérias-primas no sistema.
Atores	Vendedor
Pré-Condições	Casos de uso nº09, nº 22 e nº27.
Requisitos de Alto Nível	Req18
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao cadastro de materiais.	
	02. Solicitar ao ator o cadastro dos materiais.
03. Cadastrar os materiais.	
	04. Armazenar no sistema os novos materiais.
Pós-Condições	N/A

3.3.1.16. Caso de Uso Nº16: Cadastrar Produtos

Nome do caso de uso	Cadastrar Produtos
Descrição	O ator deseja cadastrar produtos no sistema.
Atores	Vendedor
Pré-Condições	Casos de uso nº09 e nº35.
Requisitos de Alto Nível	Req17
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao cadastro de produtos.	
	02. Solicitar ao ator o cadastro dos produtos.
03. Cadastrar os produtos.	
	05. Armazenar no sistema os novos.
Pós-Condições	N/A

3.3.1.17. Caso de Uso Nº17: Entregar Pedido

Nome do caso de uso	Entregar Pedido
Descrição	O ator deseja informar ao sistema sobre a entrega do pedido.
Atores	Vendedor
Pré-Condições	Casos de uso nº03 e nº09.

Requisitos de Alto Nível	Req22
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao processo de pedidos de compra.	
	02. Apresentar ao ator o estado atual dos pedidos de compra efetuados no sistema.
03. Selecionar o pedido de compra desejado.	
04. Registrar os dados referentes a entrega do pedido.	
	05. Armazenar no sistema os dados referentes a entrega do pedido.
Pós-Condições	N/A

3.3.1.18. Caso de Uso Nº18: Avaliar Pedido

Nome do caso de uso	Avaliar Pedido
Descrição	O ator avalia o estado de um pedido de compra de produto.
Atores	Vendedor
Pré-Condições	Casos de uso nº03 e nº09.
Requisitos de Alto Nível	Req15
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao processo de pedidos de compra.	
	02. Apresentar ao ator o estado atual dos pedidos de compra efetuados no sistema.
03. Selecionar o pedido de compra desejado.	
04. Escolher o estado atual do pedido de compra.	
	05. Armazenar no sistema o estado do pedido de compra.
Pós-Condições	N/A

3.3.1.19. Caso de Uso Nº19: Simular Compra

Nome do caso de uso	Atualizar Compra
Descrição	O ator deseja simular a compra de um produto.
Atores	Visitante
Pré-Condições	N/A
Requisitos de Alto Nível	Req03
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Selecionar a opção referente ao processo de simulação de compra.	
	02. Apresentar ao ator os produtos cadastrados no sistema com seus respectivos preços básicos.
03. Selecionar o produto desejado.	

	04. Apresentar ao ator as possibilidades de personalizações desse produto.
05. Escolher as personalizações desejadas para o produto.	
	06. Atualizar o preço do produto de acordo com as personalizações escolhidas.
	07. Apresentar ao ator a possibilidade de escolher a quantidade de produtos desejados.
08. Escolher a quantidade de produtos desejados.	
	09. Atualizar o preço final, de acordo com as alterações na quantidade de produtos.
	10. Informar ao cliente que apenas clientes cadastrados podem prosseguir com a compra.
Pós-Condições	N/A

3.3.1.20. Caso de Uso Nº20: Cadastrar Vendedor

Nome do caso de uso	Cadastrar Vendedor	
Descrição	O ator deseja cadastrar um vendedor.	
Atores	Administrador	
Pré-Condições	Caso de uso nº09	
Requisitos de Alto Nível	Req13	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Acessar a tela de cadastro de vendedores.		
		02. Solicitar as informações necessárias para o cadastro do vendedor.
03. Preencher o cadastro com os dados corretos.		
		04. Validar os dados do cadastro. *(EX1)*
		05. Verificar se o login solicitado já existe na base de dados do sistema. *(EX2)*
		06. Cadastrar os dados do vendedor na base de dados.
Fluxo de Exceção – EX1 – Cadastro preenchido com dados incorretos.		
Ações do Ator		Ações do Sistema
		01. Informar o erro e solicitar um novo preenchimento do cadastro.
02. Preencher o cadastro com os dados corretos. *(FB03)*		
Fluxo de Exceção – EX2 – Login previamente cadastrado no sistema.		
Ações do Ator		Ações do Sistema
		01. Informar ao ator que o login já existe na base e solicitar um novo login.
02. Criar um login inexistente na base do sistema. *(FB03)*		
Pós-Condições	Caso de uso nº05.	

3.3.1.21. Caso de Uso Nº21: Cadastrar Administrador

Nome do caso de uso	Cadastrar Administrador
Descrição	O ator deseja cadastrar um administrador.
Atores	Administrador
Pré-Condições	Caso de uso nº09
Requisitos de Alto Nível	Req13
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de cadastro de administradores.	
	02. Solicitar as informações necessárias para o cadastro do administrador.
03. Preencher o cadastro com os dados corretos.	
	04. Validar os dados do cadastro. *(EX1)*
	05. Verificar se o login solicitado já existe na base de dados do sistema. *(EX2)*
	06. Cadastrar os dados do vendedor na base de dados.
Fluxo de Exceção – EX1 – Cadastro preenchido com dados incorretos.	
Ações do Ator	Ações do Sistema
	01. Informar o erro e solicitar um novo preenchimento do cadastro.
02. Preencher o cadastro com os dados corretos. *(FB03)*	
Fluxo de Exceção – EX2 – Login previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o login já existe na base e solicitar um novo login.
02. Criar um login inexistente na base do sistema. *(FB03)*	
Pós-Condições	Caso de uso nº05.

3.3.1.22. Caso de Uso Nº22: Cadastrar Fornecedor

Nome do caso de uso	Cadastrar Fornecedor
Descrição	O ator deseja cadastrar um fornecedor.
Atores	Administrador
Pré-Condições	Caso de uso nº09
Requisitos de Alto Nível	Req13
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de cadastro de fornecedores.	
	02. Solicitar as informações necessárias para o cadastro do fornecedor.
03. Preencher o cadastro com os dados corretos.	
	04. Validar os dados do cadastro. *(EX1)*
	05. Verificar se o login solicitado já existe na base de dados do sistema. *(EX2)*

	06. Cadastrar os dados do fornecedor na base de dados.
Fluxo de Exceção – EX1 – Cadastro preenchido com dados incorretos.	
Ações do Ator	Ações do Sistema
	01. Informar o erro e solicitar um novo preenchimento do cadastro.
02. Preencher o cadastro com os dados corretos. *(FB03)*	
Fluxo de Exceção – EX2 – Login previamente cadastrado no sistema.	
Ações do Ator	Ações do Sistema
	01. Informar ao ator que o login já existe na base e solicitar um novo login.
02. Criar um login inexistente na base do sistema. *(FB03)*	
Pós-Condições	Caso de uso nº05.

3.3.1.23. Caso de Uso Nº23: Desativar Vendedor

Nome do caso de uso	Desativar Vendedor
Descrição	O ator deseja desativar o cadastro de um vendedor.
Atores	Administrador
Pré-Condições	Caso de uso nº20.
Requisitos de Alto Nível	Req24
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de desativação de cadastros.	
	02. Apresentar ao ator os cadastros existentes no sistema.
03. Selecionar o cadastro que deseja desativar.	
04. Solicitar a desativação do cadastro.	
	05. Solicitar confirmação da desativação.
06. Confirmar a desativação do cadastro. (sim ou não) *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Ator confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07a. Marcar o registro desativado no banco de dados do sistema.
(b)Fluxo Alternativo – AL2 – Ator não confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07b. Voltar ao passo 02 do Fluxo Básico.
Pós-Condições	N/A

3.3.1.24. Caso de Uso Nº24: Desativar Administrador

Nome do caso de uso	Desativar Administrador
Descrição	O ator deseja desativar o cadastro de um administrador.
Atores	Administrador
Pré-Condições	Caso de uso nº21.

Requisitos de Alto Nível	Req24
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de desativação de cadastros.	
	02. Apresentar ao ator os cadastros existentes no sistema.
03. Selecionar o cadastro que deseja desativar.	
04. Solicitar a desativação do cadastro.	
	05. Solicitar confirmação da desativação.
06. Confirmar a desativação do cadastro. (sim ou não) *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Ator confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07a. Marcar o registro desativado no banco de dados do sistema.
(b)Fluxo Alternativo – AL2 – Ator não confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07b. Voltar ao passo 02 do Fluxo Básico.
Pós-Condições	N/A

3.3.1.25. Caso de Uso Nº25: Desativar Fornecedor

Nome do caso de uso	Desativar Fornecedor
Descrição	O ator deseja desativar o cadastro de um fornecedor.
Atores	Administrador
Pré-Condições	Caso de uso nº22.
Requisitos de Alto Nível	Req24
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de desativação de cadastros.	
	02. Apresentar ao ator os cadastros existentes no sistema.
03. Selecionar o cadastro que deseja desativar.	
04. Solicitar a desativação do cadastro.	
	05. Solicitar confirmação da desativação.
06. Confirmar a desativação do cadastro. (sim ou não) *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Ator confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07a. Marcar o registro desativado no banco de dados do sistema.
(b)Fluxo Alternativo – AL2 – Ator não confirmou a desativação do cadastro.	
Ações do Ator	Ações do Sistema
	07b. Voltar ao passo 02 do Fluxo Básico.
Pós-Condições	N/A

3.3.1.26. Caso de Uso Nº26: Desativar Cliente

Nome do caso de uso	Desativar Cliente	
Descrição	O ator deseja desativar o cadastro de um cliente.	
Atores	Administrador	
Pré-Condições	Caso de uso nº08.	
Requisitos de Alto Nível	Req24	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Acessar a tela de desativação de cadastros.		
		02. Apresentar ao ator os cadastros existentes no sistema.
03. Selecionar o cadastro que deseja desativar.		
04. Solicitar a desativação do cadastro.		
		05. Solicitar confirmação da desativação.
06. Confirmar a desativação do cadastro. (sim ou não) *(AL1)(AL2)*		
(a)Fluxo Alternativo – AL1 – Ator confirmou a desativação do cadastro.		
Ações do Ator		Ações do Sistema
		07a. Marcar o registro desativado no banco de dados do sistema.
(b)Fluxo Alternativo – AL2 – Ator não confirmou a desativação do cadastro.		
Ações do Ator		Ações do Sistema
		07b. Voltar ao passo 02 do Fluxo Básico.
Pós-Condições	N/A	

3.3.1.27. Caso de Uso Nº27: Manter Tipos de Materiais

Nome do caso de uso	Manter Tipo de Material	
Descrição	O ator deseja administrar os tipos de materiais cadastrados no sistema.	
Atores	Administrador	
Pré-Condições	Caso de uso nº09.	
Requisitos de Alto Nível	Req21	
Fluxo Básico		
Ações do Ator		Ações do Sistema
01. Acessar a tela de manutenção dos tipos de materiais.		
		02. Apresentar a tela de manutenção de tipos de materiais.
03. Selecionar entre a alteração de um tipo de material já cadastrado ou a inclusão de um novo tipo de material. *(AL1)(AL2)*		
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de material já cadastrado.		
Ações do Ator		Ações do Sistema
		04a. Apresentar os tipos de materiais cadastrados no sistema.
05a. Selecionar o tipo de material que pretende alterar.		

06a. Alterar os dados do tipo de material desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de material no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de material.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de material.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de material no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.28. Caso de Uso Nº28: Manter Tipos de Avaliações

Nome do caso de uso	Manter Tipos de Avaliações
Descrição	O ator deseja administrar os tipos de avaliações cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de avaliações.	
	02. Apresentar a tela de manutenção de tipos de avaliações.
03. Selecionar entre a alteração de um tipo de avaliação já cadastrado ou a inclusão de um novo tipo de avaliação. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de avaliação já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de avaliações cadastrados no sistema.
05a. Selecionar o tipo de avaliação que pretende alterar.	
06a. Alterar os dados do tipo de avaliação desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de avaliação no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)

(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de avaliação.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de avaliação.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de avaliação no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.29. Caso de Uso Nº29: Manter Tipos de Condições de Pagamento

Nome do caso de uso	Manter Tipos de Condições de Pagamento
Descrição	O ator deseja administrar os tipos de condições de pagamento cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de condições de pagamento.	
	02. Apresentar a tela de manutenção de tipos de condições de pagamento.
03. Selecionar entre a alteração de um tipo de condição de pagamento já cadastrado ou a inclusão de um novo tipo de condição de pagamento. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de condição de pagamento já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de condições de pagamento cadastrados no sistema.
05a. Selecionar o tipo de condição de pagamento que pretende alterar.	
06a. Alterar os dados do tipo de condição de pagamento desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de condição de pagamento no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de avaliação.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de condição de pagamento.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de condição de pagamento no banco de dados do sistema.

Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.30. Caso de Uso Nº30: Manter Tipos de Contatos

Nome do caso de uso	Manter Tipos de Contatos
Descrição	O ator deseja administrar os tipos de contatos cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de contatos.	
	02. Apresentar a tela de manutenção de tipos de contatos.
03. Selecionar entre a alteração de um tipo de contato já cadastrado ou a inclusão de um novo tipo de contato. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de contato já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de contatos cadastrados no sistema.
05a. Selecionar o tipo de contato que pretende alterar.	
06a. Alterar os dados do tipo de contato desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de contato no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de avaliação.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de contato.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de contato no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.31. Caso de Uso Nº31: Manter Tipos de Pessoas

Nome do caso de uso	Manter Tipos de Pessoas
Descrição	O ator deseja administrar os tipos de pessoas cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de pessoas.	
	02. Apresentar a tela de manutenção de tipos de pessoas.
03. Selecionar entre a alteração de um tipo de pessoa já cadastrado ou a inclusão de um novo tipo de pessoa. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de pessoa já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de pessoas cadastrados no sistema.
05a. Selecionar o tipo de pessoa que pretende alterar.	
06a. Alterar os dados do tipo de pessoa desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de pessoa no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de pessoa.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de pessoa.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de pessoa no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.32. Caso de Uso Nº32: Manter Tipos de Credibilidades

Nome do caso de uso	Manter Tipos de Credibilidades
Descrição	O ator deseja administrar os tipos de credibilidades cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema

01. Acessar a tela de manutenção dos tipos de credibilidades.	
	02. Apresentar a tela de manutenção de tipos de credibilidades.
03. Selecionar entre a alteração de um tipo de credibilidade já cadastrado ou a inclusão de um novo tipo de credibilidade. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de credibilidade já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de credibilidades cadastrados no sistema.
05a. Selecionar o tipo de credibilidade que pretende alterar.	
06a. Alterar os dados do tipo de credibilidade desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de credibilidade no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de credibilidade.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de credibilidade.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de credibilidade no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.33. Caso de Uso Nº33: Manter Tipos de Personalizações

Nome do caso de uso	Manter Tipos de Personalizações
Descrição	O ator deseja administrar os tipos de personalizações cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de personalizações.	
	02. Apresentar a tela de manutenção de tipos de personalizações.
03. Selecionar entre a alteração de um tipo de personalização já cadastrado ou a inclusão de um novo tipo de personalização. *(AL1)(AL2)*	

(a)Fluxo Alternativo – AL1 – Alteração de um tipo de personalização já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de personalizações cadastrados no sistema.
05a. Selecionar o tipo de personalização que pretende alterar.	
06a. Alterar os dados do tipo de personalização desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de personalização no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de credibilidade.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de personalização.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de personalização no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.34. Caso de Uso Nº34: Manter Tipos de Fidelidades

Nome do caso de uso	Manter Tipos de Fidelidades
Descrição	O ator deseja administrar os tipos de fidelidades cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de fidelidades.	
	02. Apresentar a tela de manutenção de tipos de fidelidades.
03. Selecionar entre a alteração de um tipo de fidelidade já cadastrado ou a inclusão de um novo tipo de fidelidade. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de fidelidade já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de fidelidades cadastrados no sistema.
05a. Selecionar o tipo de fidelidade que pretende alterar.	
06a. Alterar os dados do tipo de fidelidade desejado.	

	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de fidelidade no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de fidelidade.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo tipo de fidelidade.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de fidelidade no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.35. Caso de Uso Nº35: Manter Tipos de Produtos

Nome do caso de uso	Manter Tipos de Produtos
Descrição	O ator deseja administrar os tipos de produtos cadastrados no sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	Req21
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção dos tipos de produtos.	
	02. Apresentar a tela de manutenção de tipos de produtos.
03. Selecionar entre a alteração de um tipo de produto já cadastrado ou a inclusão de um novo tipo de produto. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um tipo de produto já cadastrado.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os tipos de produtos cadastrados no sistema.
05a. Selecionar o tipo de produto que pretende alterar.	
06a. Alterar os dados do tipo de produto desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do tipo de produto no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo tipo de produto.	
Ações do Ator	Ações do Sistema

04b. Incluir os dados do novo tipo de produto.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo tipo de produto no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.36. Caso de Uso Nº36: Manter UF

Nome do caso de uso	Manter UF
Descrição	O ator deseja administrar a lista de estados brasileiros possíveis de cadastro pelo sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção de UF.	
	02. Apresentar a tela de manutenção de UF.
03. Selecionar entre a alteração de uma UF já cadastrada ou a inclusão de uma nova UF. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de uma UF já cadastrada.	
Ações do Ator	Ações do Sistema
	04a. Apresentar as UF cadastrados no sistema.
05a. Selecionar a UF que pretende alterar.	
06a. Alterar os dados da UF desejada.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados da UF no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de uma nova UF.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados da nova UF.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados da nova UF no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

3.3.1.37. Caso de Uso Nº37: Manter Países

Nome do caso de uso	Manter Países
Descrição	O ator deseja administrar a lista de países possíveis de cadastro pelo sistema.
Atores	Administrador
Pré-Condições	Caso de uso nº09.
Requisitos de Alto Nível	N/A
Fluxo Básico	
Ações do Ator	Ações do Sistema
01. Acessar a tela de manutenção de países.	
	02. Apresentar a tela de manutenção de países.
03. Selecionar entre a alteração de um país já cadastrado ou a inclusão de um novo país. *(AL1)(AL2)*	
(a)Fluxo Alternativo – AL1 – Alteração de um país já cadastrada.	
Ações do Ator	Ações do Sistema
	04a. Apresentar os países cadastrados no sistema.
05a. Selecionar o país que pretende alterar.	
06a. Alterar os dados do país desejado.	
	07a. Validar os dados recém cadastrados. *(EX1a)*
	08a. Armazenar os novos dados do país no banco de dados do sistema.
Fluxo de Exceção – EX1a – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04a)
(b)Fluxo Alternativo – AL2 – Inclusão de um novo país.	
Ações do Ator	Ações do Sistema
04b. Incluir os dados do novo país.	
	05b. Validar os dados recém cadastrados. *(EX1b)*
	06b. Armazenar os dados do novo país no banco de dados do sistema.
Fluxo de Exceção – EX1b – Dados preenchidos incorretamente.	
Ações do Ator	Ações do Sistema
	01. Informar o erro. (AL04b)
Pós-Condições	N/A

*2. **Diagramas de Classes***

O Diagrama de Classes é o diagrama mais utilizado e o mais importante da UML, servindo de apoio para a maioria dos outros diagramas. Ele define a estrutura das classes utilizadas pelo sistema, determinando atributos e métodos possuídos por cada classe, além de estabelecer como as classes se relacionam e trocam informações entre si.

O Diagrama de Classes apresentado na Ilustração 7 foi feito na fase de análise do projeto e visa dar uma visão do que será o sistema ainda na fase de análise do projeto.

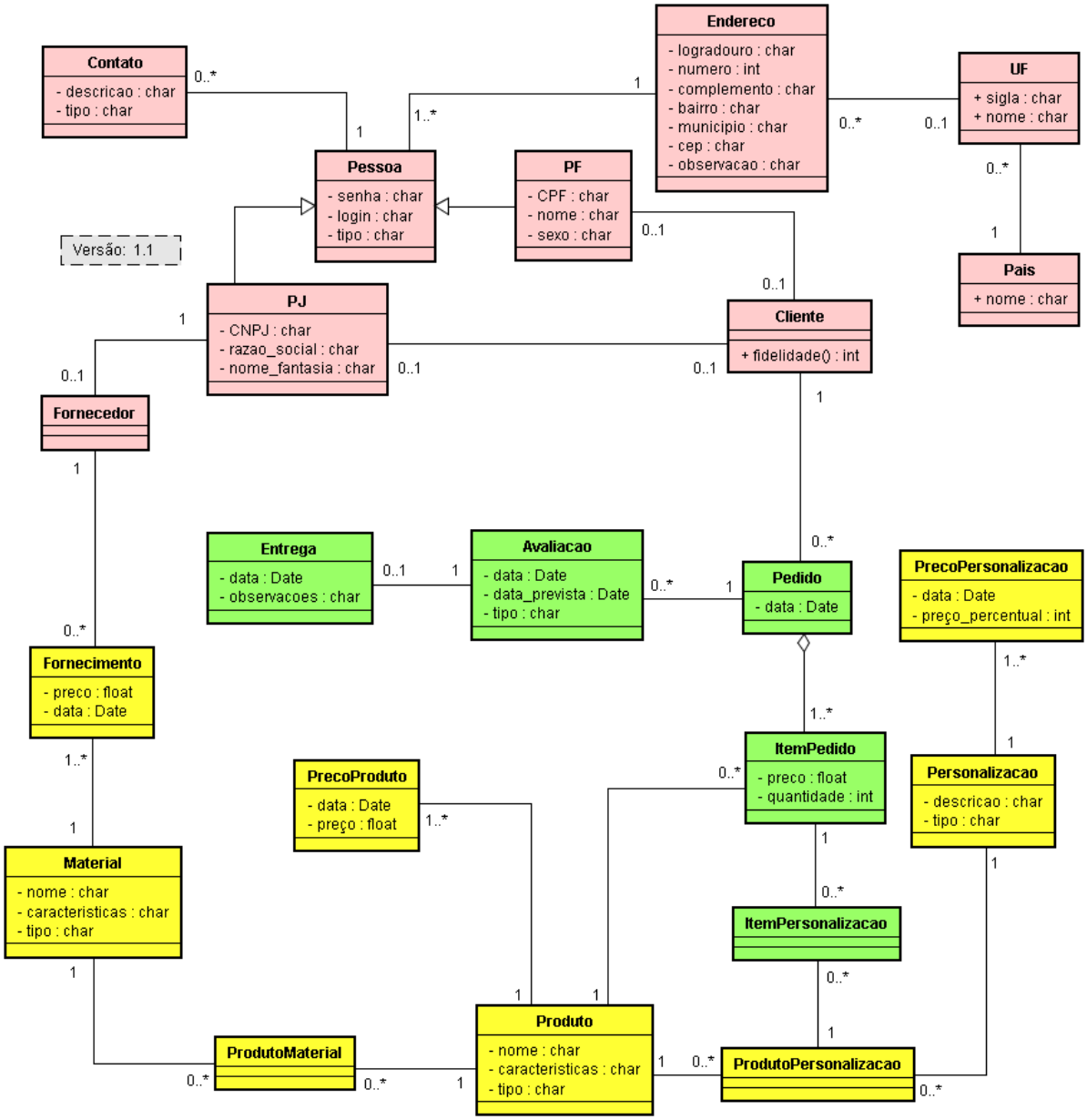

Ilustração 7 – Diagrama de Classes – Análise

3. Diagrama de Comunicação

O Diagrama de Comunicação era conhecido como Diagrama de Colaboração até a versão 1.5 da UML, tendo seu nome alterado para Diagrama de Comunicação a partir da versão 2.0 da UML.

Os diagramas de comunicação, assim como os diagramas de seqüência, são utilizados na UML para a modelagem de aspectos dinâmicos de sistemas. Eles mostram uma interação formada por um conjunto de objetos e seus relacionamentos, incluindo mensagens que

poderão ser enviadas entre eles. O diagrama de comunicação dá ênfase à organização dos objetos que participam de uma interação.

Para o sistema desenvolvido, foram criados dois diagramas de comunicação para representar a etapa de compra de um produto, pois essa é a fase mais emblemática do sistema. O primeiro diagrama de comunicação, Ilustração 8, mostra o processo de compra sob o aspecto do cliente, enquanto que no segundo diagrama, Ilustração 9, o ponto de vista focado é o do vendedor.

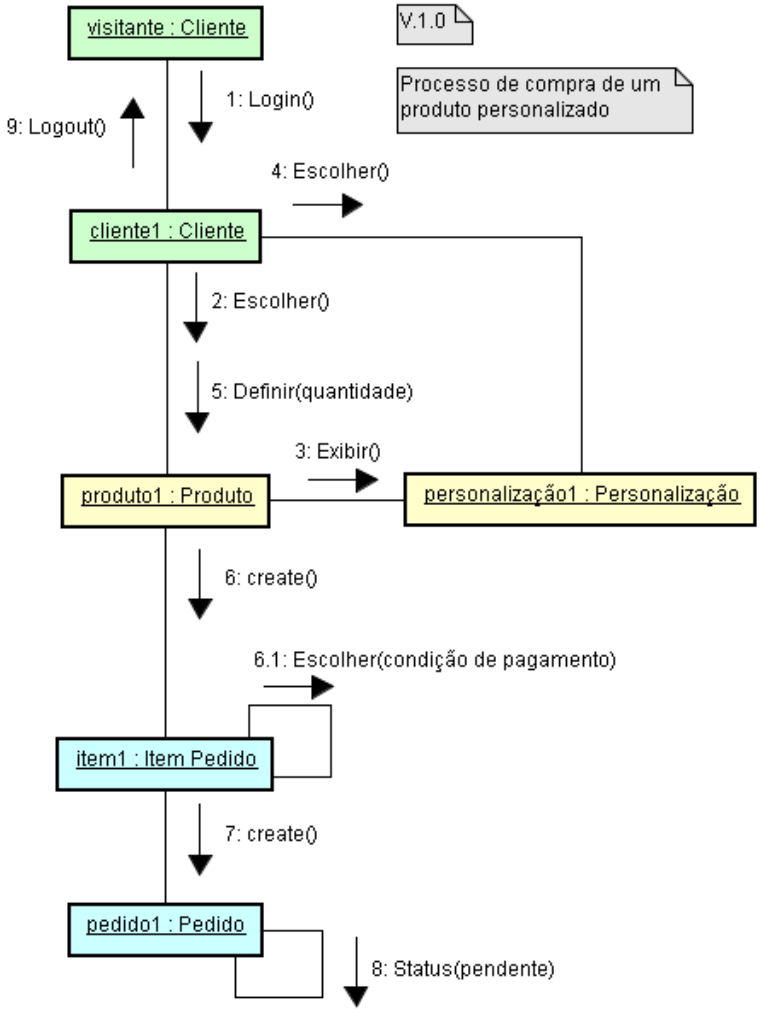

Ilustração 8 – Diagrama de Comunicação – Pedido / Cliente

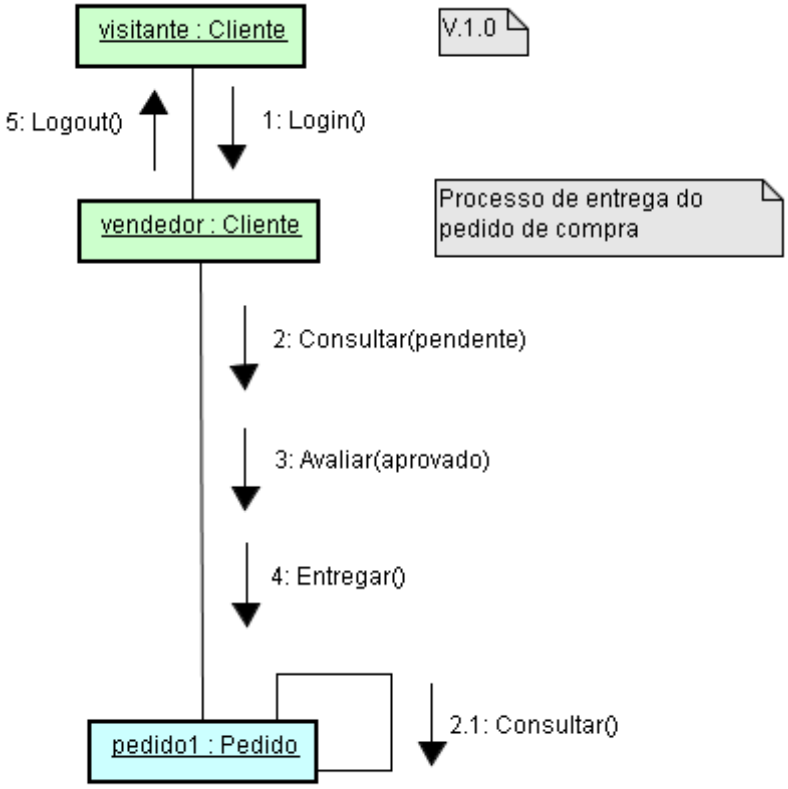

Ilustração 9 – Diagrama de Comunicação – Pedido / Vendedor

4. MODELAGEM COM UML – PROJETO

4.1. Diagrama de Classes

O diagrama de classes, na fase de projeto é mais detalhado e recebe componentes novos. Em relação aos atributos das classes os seus tipos agora são informados.

O diagrama de classes, da fase de projeto do sistema pode ser visto na Ilustração 10.

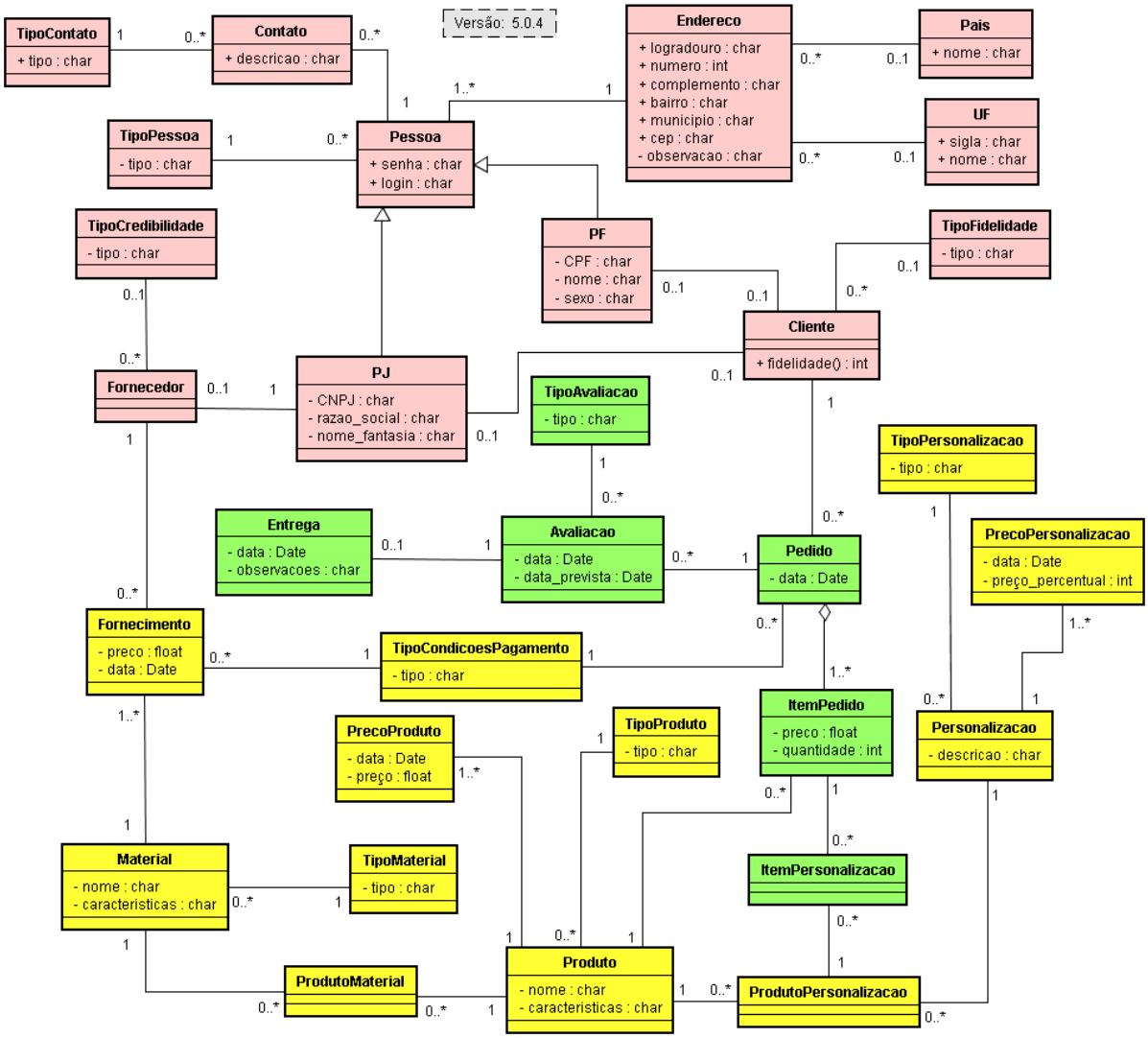

Ilustração 10 – Diagrama de Classes – Projeto

4.2. Diagrama de Componentes

O diagrama de componentes está amplamente associado à linguagem de programação utilizada para desenvolver o sistema modelado. Esse diagrama representa os componentes do sistema quando este for implantado em termos de módulos de código-fonte, bibliotecas,

formulários, arquivos de ajuda, módulos executáveis etc. e determina como esses componentes estarão estruturados e interagirão para que o sistema funcione de maneira adequada.

O diagrama de componentes do sistema pode ser visto na Ilustração 11.

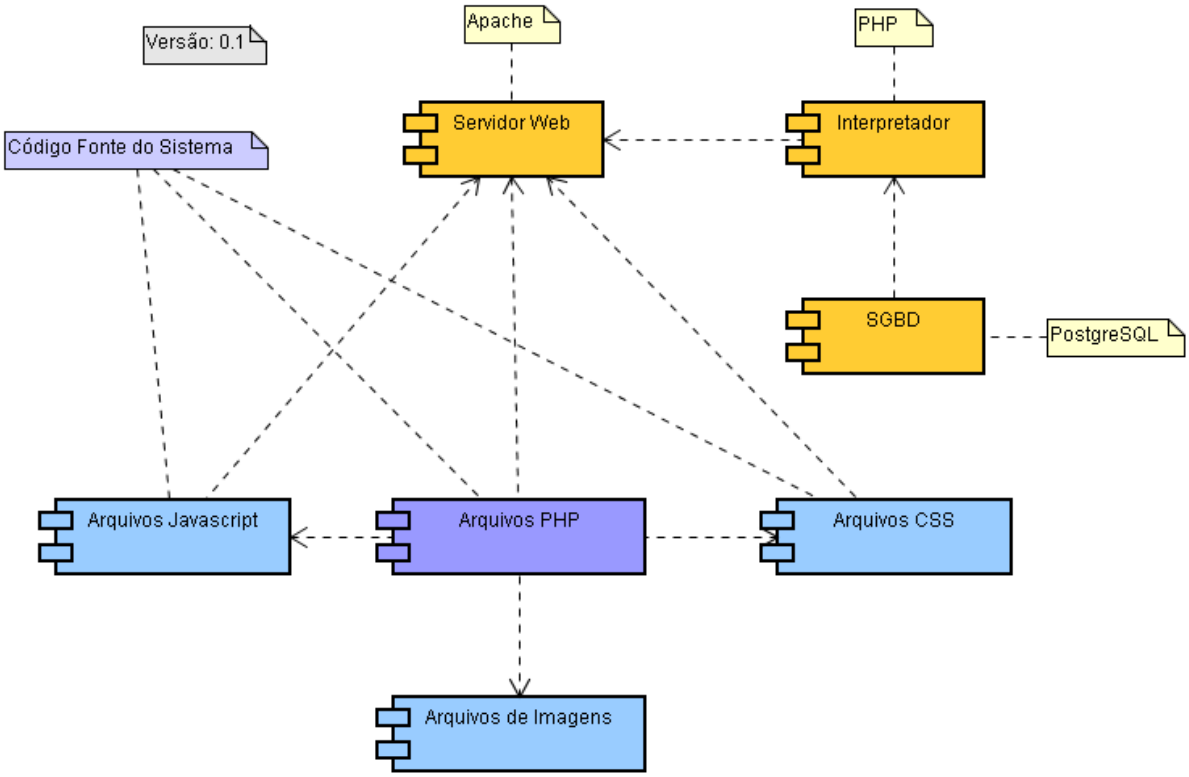

Ilustração 11 – Diagrama de componentes

4.3. Diagrama de Implantação

O diagrama de implantação determina as necessidades de hardware do sistema, as características físicas como servidores, estações, topologias e protocolos de comunicação, ou seja, todo o aspecto físico sobre o qual o sistema deverá ser executado.

O diagrama de implantação do sistema é mostrado na Ilustração 12.

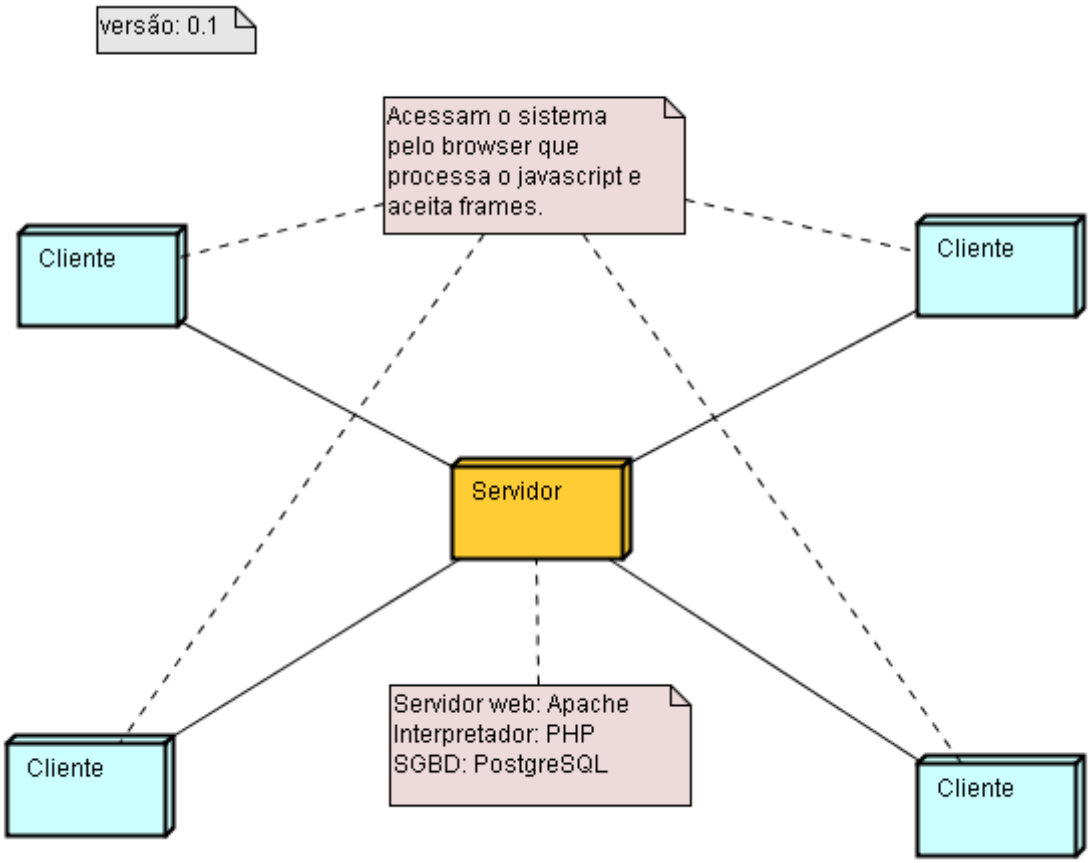

Ilustração 12 – Diagrama de Implantação

5. MODELAGEM DO BANCO DE DADOS

A modelagem de dados é a etapa responsável por informar quais são os principais objetos de dados a serem processados pelo sistema, qual a composição de cada objeto de dados e que atributos descrevem o objeto, bem como onde os objetos costumam ficar e quais relações os objetos têm entre si e entre os processos que os transformam.

5.1. Modelo Físico

O modelo físico descreve os detalhes de armazenamento (interno) dos dados e das formas de acesso a esses dados. A partir dele, é possívelvisualizar toda a estruturação do banco de dados.

O modelo físico do banco de dados do sistema está representado na Ilustração 13.

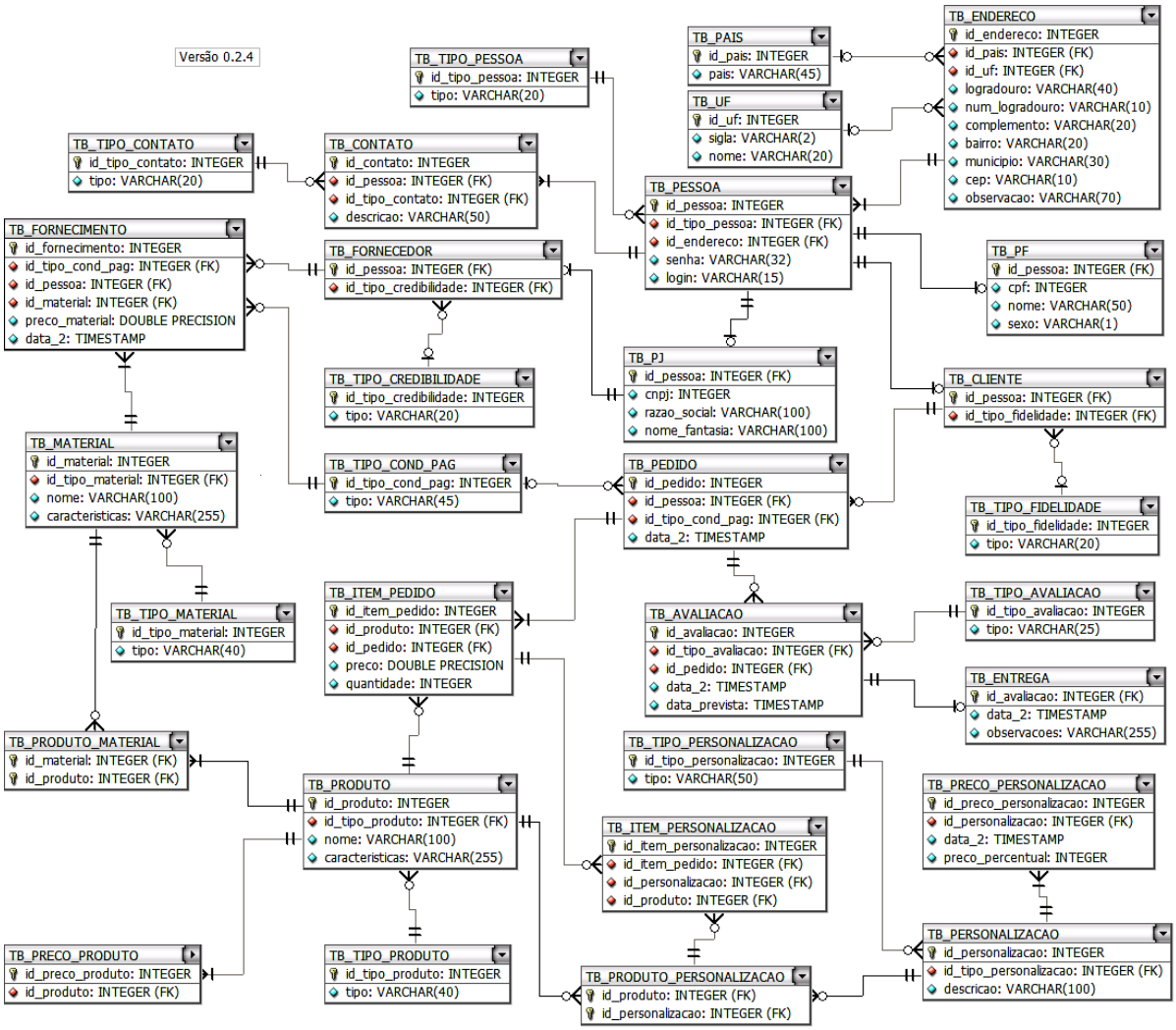

Ilustração 13 – Modelo Físico

5.2. Tabelas

Tabela 1 - Tabelas do Banco de Dados

	TABELA	*DESCRIÇÃO*
01	TB_AVALIACAO	Armazena informações sobre as avaliações feitas pelo vendedor a respeito dos pedidos de compra efetuados pelos clientes.
02	TB_CLIENTE	Armazena as informações sobre o cliente e sua fidelidade.
03	TB_CONTATO	Armazena o valor de um determinado tipo de contato de uma pessoa.
04	TB_ENDERECO	Armazena as principais informações sobre o endereço de uma pessoa.
05	TB_ENTREGA	Armazena as informações referentes às entregas dos produtos para os clientes após seus respectivos pedidos de compra terem sido aceitos.
06	TB_FORNECEDOR	Armazena informações referentes ao grau de confiança no fornecedor. Essa informação é passada pelo vendedor.
07	TB_FORNECIMENTO	Armazena as relações entre o fornecedor e o material por ele vendido, bem como as informações sobre a venda do material.
08	TB_ITEM_PEDIDO	Armazena as informações de cada item do pedido de compra.
09	TB_ITEM_PERSONALIZACAO	Armazena as informações sobre as personalizações efetuadas em cada item do pedido de compra.
10	TB_MATERIAL	Armazena as informações sobre os materiais que serão usados na composição dos produtos.
11	TB_PAIS	Armazena os países dos endereços das pessoas.
12	TB_PEDIDO	Armazena as informações sobre o pedido de compra de produto(s) feito pelo cliente.
13	TB_PERSONALIZACAO	Armazena o valor de um determinado tipo de personalização de um produto. Exemplo: armazena o valor do tamanho ou da cor do forro do produto.
14	TB_PESSOA	Armazena as informações referentes às pessoas que acessam o sistema, sejam elas PF ou PJ e atuem como vendedor, fornecedor, cliente ou administrador.
15	TB_PF	Armazena informações específicas do usuário do sistema que é uma pessoa física.
16	TB_PJ	Armazena informações específicas do usuário do sistema que é uma pessoa jurídica.
17	TB_PRECO_PERSONALIZACAO	Armazena as informações sobre os preços das personalizações oferecidas no sistema. Esse preço incide percentualmente sobre o preço do produto personalizado.
18	TB_PRECO_PRODUTO	Armazena as informações sobre os preços dos produtos oferecidos no sistema antes de serem personalizados.
19	TB_PRODUTO	Armazena informações sobre os produtos disponibilizados no sítio.
20	TB_PRODUTO_MATERIAL	Armazena as relações existentes entre os produtos e os

		materiais que o compõem.
21	TB_PRODUTO_PERSONALIZACAO	Armazena as relações existentes entre os produtos e as personalizações possíveis para eles.
22	TB_TIPO_AVALIACAO	Armazena os possíveis tipos de avaliações que podem ser dados pelo vendedor como resposta aos pedidos de compra. Exemplo: pendente, aprovado, rejeitado etc.
23	TB_TIPO_COND_PAG	Armazena os possíveis tipos de condições de pagamento para as aquisições de materiais do fornecedor. Exemplo: à vista, 2 vezes, 3 vezes etc.
24	TB_TIPO_CONTATO	Armazena os tipos de contato possíveis para uma pessoa. Exemplo: telefone, celular, fax, e-mail etc.
25	TB_TIPO_CREDIBILIDADE	Armazena os possíveis tipos de graus de credibilidade que podem ser dados ao fornecedor. Exemplo: muito confiável, confiável, desconhecido, pouco confiável etc.
26	TB_TIPO_FIDELIDADE	Armazena os possíveis tipos de graus de fidelidade que podem ser atribuídos aos clientes. Exemplo: ótimo, bom, novato etc.
27	TB_TIPO_MATERIAL	Armazena os tipos de materiais disponíveis à venda pelos fornecedores. Exemplo: prego, madeira, folha de madeira, fórmica etc.
28	TB_TIPO_PERSONALIZACAO	Armazena os tipos de personalização possíveis para os produtos. Exemplo: cor do forro, cores da madeira, com ou sem puxador etc.
29	TB_TIPO_PESSOA	Armazena os possíveis tipos de usuários que podem usufruir do sistema. Exemplo: cliente, vendedor, administrador, fornecedor etc.
30	TB_TIPO_PRODUTO	Armazena os tipos de produtos existentes. Exemplo: mesa de xadrez, caixa, bandeja, porta vinho, quadro etc.
31	TB_UF	Armazena os dados referentes às unidades federativas do Brasil.

*1. **Tabela TB_AVALIACAO***

Armazena informações sobre as avaliações feitas pelo vendedor a respeito dos pedidos de compra efetuados pelos clientes.

5.2.1.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_avaliacao	serial; unique	Esta coluna é preenchida automaticamente cada vez que uma nova avaliação for criada.
	id_tipo_avaliacao	integer; not null	Chave estrangeira originada da tabela TB_TIPO_AVALIACAO. Esse campo é o identificador do tipo de avaliação do pedido de compra. Exemplo: “Pendente”, “Aprovado”,

			”Rejeitado” etc.
	id_pedido	integer; not null	Chave estrangeira originada da tabela TB_PEDIDO. Esse campo é o identificador do pedido de compra.
	data	timestamp	Data na qual o vendedor realizou a avaliação do pedido de compra.
	data_prevista	timestamp	Data que o vendedor informou como sendo a data prevista para a entrega do pedido.

5.2.1.2. Definição

```
CREATE TABLE "TB_AVALIACAO" (
  "id_avaliacao" SERIAL UNIQUE,
  "id_pedido" INTEGER NOT NULL,
  "id_tipo_avaliacao" INTEGER NOT NULL,
  "data" TIMESTAMP NULL,
  "data_prevista" TIMESTAMP NULL,
  PRIMARY KEY ("id_avaliacao")
);
CREATE INDEX "TB_AVALIACAO_FKIndex1" ON "TB_AVALIACAO"("id_pedido");
CREATE INDEX "TB_AVALIACAO_FKIndex2" ON
"TB_AVALIACAO"("id_tipo_avaliacao");
ALTER TABLE "TB_AVALIACAO" ADD CONSTRAINT "id_pedido" FOREIGN KEY
("id_pedido") REFERENCES "TB_PEDIDO" ("id_pedido")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_AVALIACAO" ADD CONSTRAINT "id_tipo_avaliacao" FOREIGN KEY
("id_tipo_avaliacao") REFERENCES "TB_TIPO_AVALIACAO" ("id_tipo_avaliacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

2. Tabela TB_CLIENTE

Armazena as informações sobre o cliente e sua fidelidade.

5.2.2.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pessoa	integer; not null	Chave de identificação da tabela TB_CLIENTE que é importada da tabela TB_PESSOA. Esse campo é o identificador do cliente.
	id_tipo_fi	integer;	Chave estrangeira originada da tabela

	delidade	not null	TB_TIPO_FIDELIDADE. Esse campo é o identificador do tipo de fidelidade atribuída ao cliente.

5.2.2.2. Definição

```
CREATE TABLE "TB_CLIENTE" (
  "id_pessoa" INTEGER NOT NULL,
  "id_tipo_fidelidade" INTEGER NULL,
  PRIMARY KEY("id_pessoa")
);
CREATE INDEX "TB_CLIENTE_FKIndex1" ON "TB_CLIENTE"("id_pessoa");
CREATE INDEX "TB_CLIENTE_FKIndex2" ON "TB_CLIENTE"("id_tipo_fidelidade");
ALTER TABLE "TB_CLIENTE" ADD CONSTRAINT "id_tipo_fidelidade" FOREIGN KEY
("id_tipo_fidelidade") REFERENCES "TB_TIPO_FIDELIDADE"
("id_tipo_fidelidade")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_CLIENTE" ADD CONSTRAINT "id_pessoa" FOREIGN KEY
("id_pessoa") REFERENCES "TB_PESSOA" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

3. Tabela TB_CONTATO

Armazena o valor de um determinado tipo de contato de uma pessoa.

5.2.3.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_contato	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo contato for criado.
	id_pessoa	integer; not null	Chave estrangeira originada da tabela TB_PESSOA. Esse campo é o identificador da pessoa ao qual se refere o contato.
	id_tipo_contato	integer; not null	Chave estrangeira originada da tabela TB_TIPO_CONTATO. Esse campo é o identificador do tipo do contato.
	descricao	varchar(50)	Valor do contato armazenado para a pessoa. Exemplo: se o tipo do contato for “Fax”, o valor desse contato poderá ser “(21)2234-5678”.

5.2.3.2. Definição

```
CREATE TABLE "TB_CONTATO" (
  "id_contato" SERIAL UNIQUE,
  "id_pessoa" INTEGER NOT NULL,
  "id_tipo_contato" INTEGER NOT NULL,
  "descricao" VARCHAR(50) NULL,
  PRIMARY KEY("id_contato")
);
CREATE INDEX "TB_CONTATO_FKIndex1" ON "TB_CONTATO"("id_tipo_contato");
CREATE INDEX "TB_CONTATO_FKIndex2" ON "TB_CONTATO"("id_pessoa");
ALTER TABLE "TB_CONTATO" ADD CONSTRAINT "id_tipo_contato" FOREIGN KEY
("id_tipo_contato") REFERENCES "TB_TIPO_CONTATO" ("id_tipo_contato")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_CONTATO" ADD CONSTRAINT "id_pessoa" FOREIGN KEY
("id_pessoa") REFERENCES "TB_PESSOA" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

4. Tabela TB_ENDERECO

Armazena as principais informações sobre o endereço de uma pessoa.

5.2.4.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_endereco	integer; not null	Esta coluna é preenchida automaticamente cada vez que um novo endereço for criado.
	id_pais	integer; not null	Chave estrangeira originada da tabela TB_PAIS. Esse campo é o identificador do país ao qual se refere o endereço.
	id_uf	integer; not null	Chave estrangeira originada da tabela TB_UF. Esse campo é o identificador da unidade federativa ao qual se refere o endereço.
	logradouro	varchar(40)	Nome do logradouro do endereço. Exemplo: “Rua Cachambi”, “Av. Rio Branco”.
	complemento	varchar(20)	Informação complementar ao endereço da pessoa. Exemplo: “casa 06 apto. 201”.
	num_logradouro	varchar(10)	Número do endereço do logradouro.
	complemento	varchar(20)	Informação complementar ao endereço da

			pessoa. Exemplo: “casa 06 apto. 201”.
	bairro	varchar(20)	Nome do bairro do endereço. Exemplo: “Cachambi”, “Fonseca”.
	municipio	varchar(30)	Nome do município do endereço. Exemplo: “Rio de Janeiro”, “Niterói”.
	cep	varchar(10)	Código de Endereço Postal do endereço da pessoa. Exemplo: “20780120”.
	observacao	varchar(70)	Observações diversas sobre a localização do endereço. Exemplo: “vila em frente a um posto de combustível”.

5.2.4.2. Definição

```
CREATE TABLE "TB_ENDERECO" (
  "id_endereco" SERIAL UNIQUE,
  "id_pais" INTEGER NULL,
  "id_uf" INTEGER NULL,
  "logradouro" VARCHAR(40) NULL,
  "num_logradouro" VARCHAR(10) NULL,
  "complemento" VARCHAR(20) NULL,
  "bairro" VARCHAR(20) NULL,
  "municipio" VARCHAR(30) NULL,
  "cep" VARCHAR(10) NULL,
  "observacao" VARCHAR(70) NULL,
  PRIMARY KEY("id_endereco")
);
CREATE INDEX "TB_ENDERECO_FKIndex1" ON "TB_ENDERECO"("id_pais");
CREATE INDEX "TB_ENDERECO_FKIndex2" ON "TB_ENDERECO"("id_uf");
ALTER TABLE "TB_ENDERECO" ADD CONSTRAINT "id_pais" FOREIGN KEY ("id_pais")
REFERENCES "TB_PAIS" ("id_pais")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_ENDERECO" ADD CONSTRAINT "id_uf" FOREIGN KEY ("id_uf")
REFERENCES "TB_UF" ("id_uf")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

*5. **Tabela TB_ENTREGA***

Armazena as informações referentes às entregas dos produtos para os clientes após seus respectivos pedidos de compra terem sido avaliados positivamente.

5.2.5.1. *Atributos*

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_avaliacao	integer; not null	Chave de identificação da tabela TB_ ENTREGA que é importada da tabela TB_AVALIACAO. Esse campo é o identificador da avaliação.
	data	timestamp	Data na qual foi feita a entrega do pedido ao cliente.
	observacoes	varchar(100)	Observações diversas sobre a entrega do produto.

5.2.5.2. *Definição*

```
CREATE TABLE "TB_ENTREGA" (
  "id_avaliacao" INTEGER NOT NULL,
  "data" TIMESTAMP NULL,
  "observacoes" VARCHAR(100) NULL,
  PRIMARY KEY("id_avaliacao")
);
CREATE INDEX "TB_ENTREGA_FKIndex1" ON "TB_ENTREGA"("id_avaliacao");
ALTER TABLE "TB_ENTREGA" ADD CONSTRAINT "id_avaliacao" FOREIGN KEY
("id_avaliacao") REFERENCES "TB_AVALIACAO" ("id_avaliacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

6. ***Tabela TB_FORNECEDOR***

Armazena informações referentes ao grau de confiança no fornecedor. Essa informação é passada pelo vendedor.

5.2.6.1. *Atributos*

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pessoa	integer; not null	Chave de identificação da tabela TB_FORNECEDOR que é importada da tabela TB_PESSOA. Esse campo é o identificador da pessoa representada como sendo um fornecedor.
	id_tipo_credibilidade	integer; not null	Chave estrangeira originada da tabela TB_TIPO_CREDIBILIDADE. Esse campo é o

			identificador do tipo de credibilidade associada aos fornecedores.

5.2.6.2. Definição

```
CREATE TABLE "TB_FORNECEDOR" (
  "id_pessoa" INTEGER NOT NULL,
  "id_tipo_credibilidade" INTEGER NULL,
  PRIMARY KEY("id_pessoa")
);
CREATE INDEX "TB_FORNECEDOR_FKIndex1" ON "TB_FORNECEDOR"("id_pessoa");
CREATE INDEX "TB_FORNECEDOR_FKIndex2" ON
"TB_FORNECEDOR"("id_tipo_credibilidade");
ALTER TABLE "TB_FORNECEDOR" ADD CONSTRAINT "id_pessoa" FOREIGN KEY
("id_pessoa") REFERENCES "TB_PJ" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_FORNECEDOR" ADD CONSTRAINT "id_tipo_credibilidade" FOREIGN
KEY ("id_tipo_credibilidade") REFERENCES "TB_TIPO_CREDIBILIDADE"
("id_tipo_credibilidade")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

7. Tabela TB_ FORNECIMENTO

Armazena as relações entre o fornecedor e o material por ele vendido, bem como as informações sobre a venda do material.

5.2.7.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_fornecimento	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo fornecimento ocorre.
	id_pessoa	integer; not null	Chave estrangeira originada da tabela TB_PESSOA. Campo identificador do fornecedor do material.
	id_material	integer; not null	Chave estrangeira originada da tabela TB_MATERIAL. Campo identificador do material fornecido pelo fornecedor.
	id_tipo_cond_pag	integer;	Chave estrangeira originada da tabela

		not null	TB_TIPO_COND_PAG. Campo identificador do tipo da condição de pagamento utilizada na compra dos materiais fornecidos pelo fornecedor.
	preco	money	Identifica o preço do material comprado.
	data	timestamp	Data na qual a compra do material foi registrada no sistema.

5.2.7.2. Definição

```
CREATE TABLE "TB_FORNECIMENTO" (
  "id_fornecimento" SERIAL UNIQUE,
  "id_pessoa" INTEGER NOT NULL,
  "id_material" INTEGER NOT NULL,
  "id_tipo_cond_pag" INTEGER NOT NULL,
  "preco" MONEY NULL,
  "data" TIMESTAMP NULL,
  PRIMARY KEY("id_fornecimento")
);
CREATE INDEX "TB_FORNECIMENTO_FKIndex1" ON
"TB_FORNECIMENTO"("id_material");
CREATE INDEX "TB_FORNECIMENTO_FKIndex2" ON "TB_FORNECIMENTO"("id_pessoa");
CREATE INDEX "TB_FORNECIMENTO_FKIndex3" ON
"TB_FORNECIMENTO"("id_tipo_cond_pag");
ALTER TABLE "TB_FORNECIMENTO" ADD CONSTRAINT "id_material" FOREIGN KEY
("id_material") REFERENCES "TB_MATERIAL" ("id_material")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_FORNECIMENTO" ADD CONSTRAINT "id_pessoa" FOREIGN KEY
("id_pessoa") REFERENCES "TB_FORNECEDOR" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_FORNECIMENTO" ADD CONSTRAINT "id_tipo_cond_pag" FOREIGN KEY
("id_tipo_cond_pag") REFERENCES "TB_TIPO_COND_PAG" ("id_tipo_cond_pag")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

*8. **Tabela TB_ ITEM_PEDIDO***

Armazena as informações de cada item do pedido de compra.

5.2.8.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_produto	integer; not null	Chave de identificação da tabela TB_ITEM_PEDIDO que é importada da tabela TB_PRODUTO. Campo identificador do produto do qual se compõe o item de pedido.
	id_produto	integer; not null	Chave estrangeira originada da tabela TB_PRODUTO. Campo identificador do produto do qual se compõe o item de pedido.
	id_pedido	integer; not null	Chave estrangeira originada da tabela TB_PEDIDO. Campo identificador do pedido de compra.
	preco	money	Preço do item referido no pedido de compra.
	quantidade	integer	Quantidade de um determinado item de pedido, dentro do pedido de compra.

5.2.8.2. Definição

```
CREATE TABLE "TB_ITEM_PEDIDO" (
  "id_item_pedido" SERIAL UNIQUE,
  "id_produto" INTEGER NOT NULL,
  "id_pedido" INTEGER NOT NULL,
  "preco" MONEY NULL,
  "quantidade" INTEGER NULL,
  PRIMARY KEY("id_item_pedido")
);
CREATE INDEX "TB_ITEM_PEDIDO_FKIndex1" ON "TB_ITEM_PEDIDO"("id_pedido");
CREATE INDEX "TB_ITEM_PEDIDO_FKIndex2" ON "TB_ITEM_PEDIDO"("id_produto");
ALTER TABLE "TB_ITEM_PEDIDO" ADD CONSTRAINT "id_pedido" FOREIGN KEY
("id_pedido") REFERENCES "TB_PEDIDO" ("id_pedido")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_ITEM_PEDIDO" ADD CONSTRAINT "id_produto" FOREIGN KEY
("id_produto") REFERENCES "TB_PRODUTO" ("id_produto")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

9. Tabela TB_ITEM_PERSONALIZACAO

Armazena as informações sobre as personalizações efetuadas em cada item do pedido de compra.

5.2.9.1. *Atributos*

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_item_personalizacao	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo item de personalização for criado.
	id_item_pedido	integer; not null	Chave estrangeira originada da tabela TB_ITEM_PEDIDO. Campo identificador do item do pedido de compra que foi personalizado.
	id_personalizacao	integer; not null	Chave estrangeira originada da tabela TB_PERSONALIZACAO. Campo identificador da personalização usado pelo item do pedido de compra.
	id_produto	integer; not null	Chave estrangeira originada da tabela TB_PRODUTO. Campo identificador do produto a que se refere o item do pedido de compra que foi personalizado.

5.2.9.2. *Definição*

```
CREATE TABLE "TB_ITEM_PERSONALIZACAO" (
  "id_item_personalizacao" SERIAL UNIQUE,
  "id_item_pedido" INTEGER NULL,
  "id_personalizacao" INTEGER NULL,
  "id_produto" INTEGER NULL,
  PRIMARY KEY("id_item_personalizacao")
);
CREATE INDEX "TB_ITEM_PERSONALIZACAO_FKIndex1" ON
"TB_ITEM_PERSONALIZACAO"("id_item_pedido");
CREATE INDEX "TB_ITEM_PERSONALIZACAO_FKIndex2" ON
"TB_ITEM_PERSONALIZACAO"("id_personalizacao","id_produto");
ALTER TABLE "TB_ITEM_PERSONALIZACAO" ADD CONSTRAINT
"id_produto_personalizacao" FOREIGN KEY ("id_produto","id_personalizacao")
REFERENCES "TB_PRODUTO_PERSONALIZACAO" ("id_produto","id_personalizacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_ITEM_PERSONALIZACAO" ADD CONSTRAINT "id_item_pedido"
FOREIGN KEY ("id_item_pedido") REFERENCES "TB_ITEM_PEDIDO"
("id_item_pedido")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

10. Tabela TB_ MATERIAL

Armazena as informações sobre os materiais que serão usados na composição dos produtos.

5.2.10.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_material	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo material for cadastrado.
	id_tipo_material	integer; not null	Chave estrangeira originada da tabela TB_TIPO_MATERIAL. Trata-se da identificação do tipo de material.
	nome	varchar(100)	Descreve o nome do material. Exemplo: “folha de cerejeira”, “cola da marca XX”
	caracteristicas	varchar(255)	Descreve as características do material.

5.2.10.2. Definição

```
CREATE TABLE "TB_MATERIAL" (
  "id_material" SERIAL UNIQUE,
  "id_tipo_material" INTEGER NOT NULL,
  "nome" VARCHAR(100) NULL,
  "caracteristicas" VARCHAR(255) NULL,
  PRIMARY KEY("id_material")
);
CREATE INDEX "TB_MATERIAL_FKIndex1" ON "TB_MATERIAL"("id_tipo_material");
ALTER TABLE "TB_MATERIAL" ADD CONSTRAINT "id_tipo_material" FOREIGN KEY
("id_tipo_material") REFERENCES "TB_TIPO_MATERIAL" ("id_tipo_material")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

11. Tabela TB_ PAIS

Armazena os países dos endereços das pessoas.

5.2.11.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pais	serial;	Esta coluna é preenchida automaticamente cada vez

		unique	que um novo país for cadastrado no sistema.
	pais	varchar(45)	Nome do país do endereço cadastrado pela pessoa.

5.2.11.2. Definição

```
CREATE TABLE "TB_PAIS" (
  "id_pais" SERIAL UNIQUE,
  "pais" VARCHAR(45) NULL,
  PRIMARY KEY("id_pais")
);
```

*12. **Tabela TB_ PEDIDO***

Armazena as informações sobre o pedido de compra de produto(s) feito pelo cliente.

5.2.12.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pedido	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo pedido de compra for criado.
	id_pessoa	integer; not null	Chave estrangeira originada da tabela TB_PESSOA. Trata-se da identificação do cliente que está efetuando o pedido de compra.
	id_tipo_cond_pag	integer; not null	Chave estrangeira originada da tabela TB_TIPO_COND_PAG. Trata-se da identificação do tipo de condição de pagamento do pedido de compra.
	data	timestamp	Informação sobre a data na qual o cliente efetuou o pedido de compra.

5.2.12.2. Definição

```
CREATE TABLE "TB_PEDIDO" (
  "id_pedido" SERIAL UNIQUE,
  "id_pessoa" INTEGER NOT NULL,
  "id_tipo_cond_pag" INTEGER NOT NULL,
  "data" TIMESTAMP NULL,
  PRIMARY KEY("id_pedido")
);
CREATE INDEX "TB_PEDIDO_FKIndex1" ON "TB_PEDIDO"("id_pessoa");
```

```
CREATE INDEX "TB_PEDIDO_FKIndex2" ON "TB_PEDIDO"("id_tipo_cond_pag");
ALTER TABLE "TB_PEDIDO" ADD CONSTRAINT "id_pessoa" FOREIGN KEY
("id_pessoa") REFERENCES "TB_CLIENTE" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_PEDIDO" ADD CONSTRAINT "id_tipo_cond_pag" FOREIGN KEY
("id_tipo_cond_pag") REFERENCES "TB_TIPO_COND_PAG" ("id_tipo_cond_pag")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

13. Tabela TB_ PERSONALIZACAO

Armazena o valor de um determinado tipo de personalização de um produto. Exemplo: armazena o valor do tamanho ou da cor do forro do produto.

5.2.13.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_personalizacao	integer; not null	Esta coluna é preenchida automaticamente cada vez que uma nova personalização for criada.
	id_tipo_personalizacao	integer; not null	Chave estrangeira originada da tabela TB_TIPO_PERSONALIZACAO. Trata-se da identificação do tipo de personalização que será feita no item de pedido.
	descricao	varchar(100)	Armazena a descrição da personalização do produto. Exemplo: se o tipo de personalização for cor de forro, esse campo poderá ser “vermelho”.

5.2.13.2. Definição

```
CREATE TABLE "TB_PERSONALIZACAO" (
  "id_personalizacao" SERIAL UNIQUE,
  "id_tipo_personalizacao" INTEGER NOT NULL,
  "descricao" VARCHAR(100) NULL,
  PRIMARY KEY("id_personalizacao")
);
CREATE INDEX "TB_PERSONALIZACAO_FKIndex1" ON
"TB_PERSONALIZACAO"("id_tipo_personalizacao");
ALTER TABLE "TB_PERSONALIZACAO" ADD CONSTRAINT "id_tipo_personalizacao"
FOREIGN KEY ("id_tipo_personalizacao") REFERENCES "TB_TIPO_PERSONALIZACAO"
```

```
("id_tipo_personalizacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

14. Tabela TB_PESSOA

Armazena as informações referentes às pessoas que acessam o sistema, sejam elas PF ou PJ e atuem como vendedor, fornecedor, cliente ou administrador.

5.2.14.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pessoa	serial; unique	Esta coluna é preenchida automaticamente a cada nova pessoa cadastrada.
	id_tipo_pessoa	integer; not null	Chave estrangeira originada da tabela TB_TIPO_PESSOA. Trata-se da identificação do tipo de pessoa que está acessando o sistema. Ex: cliente, fornecedor, vendedor ou administrador.
	id_endereco	integer	Chave estrangeira originada da tabela TB_ENDERECO. Trata-se da identificação do endereço da pessoa.
	senha	varchar(32)	Senha usada na identificação de cada pessoa no sistema.
	login	varchar(15)	Login de identificação de cada pessoa no sistema.

5.2.14.2. Definição

```
CREATE TABLE "TB_PESSOA" (
  "id_pessoa" SERIAL UNIQUE,
  "id_tipo_pessoa" INTEGER NOT NULL,
  "id_endereco" INTEGER NULL,
  "senha" VARCHAR(32) NULL,
  "login" VARCHAR(15) NULL,
  PRIMARY KEY("id_pessoa")
);
CREATE INDEX "TB_PESSOA_FKIndex1" ON "TB_PESSOA"("id_endereco");
CREATE INDEX "TB_PESSOA_FKIndex2" ON "TB_PESSOA"("id_tipo_pessoa");
ALTER TABLE "TB_PESSOA" ADD CONSTRAINT "id_endereco" FOREIGN KEY
("id_endereco") REFERENCES "TB_ENDERECO" ("id_endereco")
```

```
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_PESSOA" ADD CONSTRAINT "id_tipo_pessoa" FOREIGN KEY
("id_tipo_pessoa") REFERENCES "TB_TIPO_PESSOA" ("id_tipo_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

15. Tabela TB_PF

Armazena informações específicas do usuário do sistema que é uma pessoa física.

5.2.15.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pessoa	integer; not null	Chave de identificação da tabela TB_PF que é importada da tabela TB_PESSOA. Esse campo é o identificador da pessoa representada como uma pessoa física.
	cpf	varchar(15)	Campo onde serão armazenados os números do CPF das pessoas físicas cadastradas no sistema.
	nome	varchar(50)	Nome completo das pessoas físicas cadastradas no sistema.
	sexo	varchar(1)	Sexo das pessoas físicas cadastradas no sistema.

5.2.15.2. Definição

```
CREATE TABLE "TB_PF" (
  "id_pessoa" INTEGER NOT NULL,
  "cpf" VARCHAR(15) NOT NULL,
  "nome" VARCHAR(50) NOT NULL,
  "sexo" VARCHAR(1) NOT NULL,
  PRIMARY KEY("id_pessoa")
);
CREATE INDEX "TB_PF_FKIndex1" ON "TB_PF"("id_pessoa");
ALTER TABLE "TB_PF" ADD CONSTRAINT "id_pessoa" FOREIGN KEY ("id_pessoa")
REFERENCES "TB_PESSOA" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

16. Tabela TB_PJ

Armazena informações específicas do usuário do sistema que é uma pessoa jurídica.

5.2.16.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_pessoa	integer; not null	Chave de identificação da tabela TB_PJ que é importada da tabela TB_PESSOA. Esse campo é o identificador da pessoa representada como uma pessoa jurídica.
	cnpj	varchar(15)	Campo onde serão armazenados os números do CNPJ das pessoas jurídicas cadastradas no sistema.
	razao_social	varchar(100)	Nome da razão social das pessoas jurídicas cadastradas no sistema.
	nome_fantasia	varchar(100)	Nome fantasia das pessoas jurídicas cadastradas no sistema.

5.2.16.2. Definição

```
CREATE TABLE "TB_PJ" (
  "id_pessoa" INTEGER NOT NULL,
  "cnpj" VARCHAR(15) NOT NULL,
  "razao_social" VARCHAR(100) NULL,
  "nome_fantasia" VARCHAR(100) NULL,
  PRIMARY KEY("id_pessoa")
);
CREATE INDEX "TB_PJ_FKIndex1" ON "TB_PJ"("id_pessoa");
ALTER TABLE "TB_PJ" ADD CONSTRAINT "id_pessoa" FOREIGN KEY ("id_pessoa")
REFERENCES "TB_PESSOA" ("id_pessoa")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

17. Tabela TB_PRECO_PERSONALIZACAO

Armazena as informações sobre os preços das personalizações oferecidas no sistema. Esse preço incide percentualmente sobre o preço do produto personalizado.

5.2.17.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_preco_personalizacao	serial; unique	Esta coluna é preenchida automaticamente cada vez que um

			novo preço de personalização for cadastrado no sistema.
	id_personalizacao	integer; not null	Chave estrangeira originada da tabela TB_TIPO_PERSONALIZACAO. Trata-se da identificação da personalização à qual se refere o preço em questão.
	data	timestamp	Data na qual o preço da personalização foi atualizado.
	preco_percentual	integer; not null	Percentual do preço do produto alterado pela personalização.

5.2.17.2. Definição

```
CREATE TABLE "TB_PRECO_PERSONALIZACAO" (
  "id_preco_personalizacao" SERIAL UNIQUE,
  "id_personalizacao" INTEGER NOT NULL,
  "data" TIMESTAMP NULL,
  "preco_percentual" INTEGER NOT NULL,
  PRIMARY KEY("id_preco_personalizacao")
);
CREATE INDEX "TB_PRECO_PERSONALIZACAO_FKIndex1" ON
"TB_PRECO_PERSONALIZACAO"("id_personalizacao");
ALTER TABLE "TB_PRECO_PERSONALIZACAO" ADD CONSTRAINT "id_personalizacao"
FOREIGN KEY ("id_personalizacao") REFERENCES "TB_PERSONALIZACAO"
("id_personalizacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

18. Tabela TB_PRECO_PRODUTO

Armazena as informações sobre os preços dos produtos oferecidos no sistema antes de serem personalizados.

5.2.18.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_preco_produto	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo preço de produto for cadastrado no

			sistema.
	id_produto	integer; not null	Chave estrangeira originada da tabela TB_PRODUTO. Esse campo é o identificador do produto cujo preço é informado nessa tabela.
	data	timestamp	Data na qual o preço do produto foi atualizado.
	preco	money	Preço do produto sem personalização.

5.2.18.2. Definição

```
CREATE TABLE "TB_PRECO_PRODUTO" (
  "id_preco_produto" SERIAL UNIQUE,
  "id_produto" INTEGER NOT NULL,
  "data" TIMESTAMP NULL,
  "preco" MONEY NULL,
  PRIMARY KEY("id_preco_produto")
);
CREATE INDEX "TB_PRECO_PRODUTO_FKIndex1" ON
"TB_PRECO_PRODUTO"("id_produto");
ALTER TABLE "TB_PRECO_PRODUTO" ADD CONSTRAINT "id_produto" FOREIGN KEY
("id_produto") REFERENCES "TB_PRODUTO" ("id_produto")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

*19. **Tabela TB_PRODUTO***

Armazena informações sobre os produtos disponibilizados no sítio.

5.2.19.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_produto	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo produto for cadastrado no sistema.
	id_tipo_produto	integer; not null	Chave estrangeira originada da tabela TB_TIPO_PRODUTO. Esse campo é o identificador do tipo do produto.
	nome	varchar(100)	Descreve o nome do produto. Exemplo: “mesa de xadrez”, “Porta Laptop Listrado”
	caracteristicas	varchar(255)	Descreve as características do produto.

5.2.19.2. Definição

```
CREATE TABLE "TB_PRODUTO" (
  "id_produto" SERIAL UNIQUE,
  "id_tipo_produto" INTEGER NOT NULL,
  "nome" VARCHAR(100) NULL,
  "caracteristicas" VARCHAR(255) NULL,
  PRIMARY KEY("id_produto")
);
CREATE INDEX "TB_PRODUTO_FKIndex1" ON "TB_PRODUTO"("id_tipo_produto");
ALTER TABLE "TB_PRODUTO" ADD CONSTRAINT "id_tipo_produto" FOREIGN KEY
("id_tipo_produto") REFERENCES "TB_TIPO_PRODUTO" (id_tipo_produto)
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

20. Tabela TB_PRODUTO_MATERIAL

Armazena as relações existentes entre os produtos e os materiais que o compõem.

5.2.20.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_material	integer; not null	Chave de identificação da tabela TB_PRODUTO_MATERIAL que é importada da tabela TB_MATERIAL. Esse campo é o identificador de cada material que compõe o produto.
	id_produto	integer; not null	Chave de identificação da tabela TB_PRODUTO_MATERIAL que é importada da tabela TB_PRODUTO. Esse campo é o identificador do produto.

5.2.20.2. Definição

```
CREATE TABLE "TB_PRODUTO_MATERIAL" (
  "id_material" INTEGER NOT NULL,
  "id_produto" INTEGER NOT NULL,
  PRIMARY KEY("id_material", "id_produto")
);
CREATE INDEX "TB_PRODUTO_MATERIAL_FKIndex1" ON
"TB_PRODUTO_MATERIAL"("id_material");
CREATE INDEX "TB_PRODUTO_MATERIAL_FKIndex2" ON
```

```
"TB_PRODUTO_MATERIAL"("id_produto");
ALTER TABLE "TB_PRODUTO_MATERIAL" ADD CONSTRAINT "id_material" FOREIGN KEY
("id_material") REFERENCES "TB_MATERIAL" ("id_material")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
ALTER TABLE "TB_PRODUTO_MATERIAL" ADD CONSTRAINT "id_produto" FOREIGN KEY
("id_produto") REFERENCES "TB_PRODUTO" ("id_produto")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

21. Tabela TB_PRODUTO_PERSONALIZACAO

Armazena as relações existentes entre os produtos e as personalizações possíveis para eles.

5.2.21.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_personalizacao	integer; not null	Coluna de identificação do tipo de avaliação. Esse campo é preenchido automaticamente para cada novo tipo de avaliação cadastrado no sistema.
	id_produto	integer; not null	Chave de identificação da tabela TB_PRODUTO_MATERIAL que é importada da tabela TB_PRODUTO. Esse campo é o identificador do produto.

5.2.21.2. Definição

```
CREATE TABLE "TB_PRODUTO_PERSONALIZACAO" (
  "id_personalizacao" INTEGER NOT NULL,
  "id_produto" INTEGER NOT NULL,
  PRIMARY KEY("id_personalizacao","id_produto")
);
CREATE INDEX "TB_PRODUTO_PERSONALIZACAO_FKIndex1" ON
"TB_PRODUTO_PERSONALIZACAO"("id_personalizacao");
CREATE INDEX "TB_PRODUTO_PERSONALIZACAO_FKIndex2" ON
"TB_PRODUTO_PERSONALIZACAO"("id_produto");
ALTER TABLE "TB_PRODUTO_PERSONALIZACAO" ADD CONSTRAINT "id_personalizacao"
FOREIGN KEY ("id_personalizacao") REFERENCES "TB_PERSONALIZACAO"
("id_personalizacao")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

```
ALTER TABLE "TB_PRODUTO_PERSONALIZACAO" ADD CONSTRAINT "id_produto" FOREIGN
KEY ("id_produto") REFERENCES "TB_PRODUTO" ("id_produto")
    ON UPDATE NO ACTION ON DELETE NO ACTION;
```

22. Tabela TB_TIPO_AVALIACAO

Armazena os possíveis tipos de avaliações que podem ser dados pelo vendedor como resposta aos pedidos de compra. Exemplo: pendente, aprovado, rejeitado etc.

5.2.22.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_avaliacao	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de avaliação for cadastrado no sistema.
	tipo	varchar(25)	Tipo de avaliação cadastrada. Exemplo: pendente, aprovado, rejeitado etc.

5.2.22.2. Definição

```
CREATE TABLE "TB_TIPO_AVALIACAO" (
  "id_tipo_avaliacao" SERIAL UNIQUE,
  "tipo" VARCHAR(25) NULL,
  PRIMARY KEY("id_tipo_avaliacao")
);
```

23. Tabela TB_TIPO_COND_PAG

Armazena os possíveis tipos de condições de pagamento para as aquisições de materiais do fornecedor. Exemplo: à vista, 2 vezes, 3 vezes etc.

5.2.23.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_cond_pag	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de condição de pagamento for cadastrado no sistema.

	tipo	varchar(45)	Tipo de condição de pagamento cadastrado. Exemplo: à vista, 2 vezes, 3 vezes etc.

5.2.23.2. *Definição*

```
CREATE TABLE "TB_TIPO_COND_PAG" (
  "id_tipo_cond_pag" SERIAL UNIQUE,
  "tipo" VARCHAR(45) NULL,
  PRIMARY KEY("id_tipo_cond_pag")
);
```

24. ***Tabela TB_TIPO_CONTATO***

Armazena os tipos de contato possíveis para uma pessoa. Exemplo: telefone, celular, fax, e-mail etc.

5.2.24.1. *Atributos*

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_contato	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de contato for cadastrado no sistema.
	tipo	varchar(20)	Tipo do contato cadastrado. Exemplo: telefone, fax, e-mail etc.

5.2.24.2. *Definição*

```
CREATE TABLE "TB_TIPO_CONTATO" (
  "id_tipo_contato" SERIAL UNIQUE,
  "tipo" VARCHAR(20) NULL,
  PRIMARY KEY("id_tipo_contato")
);
```

25. ***Tabela TB_TIPO_CREDIBILIDADE***

Armazena os possíveis tipos de graus de credibilidade que podem ser dados ao fornecedor. Exemplo: muito confiável, confiável, desconhecido, pouco confiável etc.

5.2.25.1. *Atributos*

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_ credibilidade	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de credibilidade for cadastrado no sistema.
	tipo	varchar(20)	Tipo de credibilidade cadastrado. Exemplo: muito confiável, confiável, desconhecido, pouco confiável etc.

5.2.25.2. Definição

```
CREATE TABLE "TB_TIPO_CREDIBILIDADE" (
  "id_tipo_credibilidade" SERIAL UNIQUE,
  "tipo" VARCHAR(20) NULL,
  PRIMARY KEY("id_tipo_credibilidade")
);
```

26. Tabela TB_TIPO_FIDELIDADE

Armazena os possíveis tipos de graus de fidelidade que podem ser atribuídos aos clientes. Exemplo: ótimo, bom, novato etc.

5.2.26.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_ fidelidade	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de fidelidade for cadastrado no sistema.
	tipo	varchar(20)	Tipo de fidelidade cadastrado. Exemplo: ótimo, bom, novato etc.

5.2.26.2. Definição

```
CREATE TABLE "TB_TIPO_FIDELIDADE" (
  "id_tipo_fidelidade" SERIAL UNIQUE,
  "tipo" VARCHAR(20) NULL,
  PRIMARY KEY("id_tipo_fidelidade")
);
```

27. Tabela TB_TIPO_MATERIAL

Armazena os tipos de materiais disponíveis à venda pelos fornecedores. Exemplo: prego, madeira, folha de madeira, fórmica etc.

5.2.27.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_material	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de material for cadastrado no sistema.
	tipo	varchar(40)	Tipo de material cadastrado. Exemplo: parafuso, prego, folha de madeira, alça etc.

5.2.27.2. Definição

```
CREATE TABLE "TB_TIPO_MATERIAL" (
  "id_tipo_material" SERIAL UNIQUE,
  "tipo" VARCHAR(40) NULL,
  PRIMARY KEY("id_tipo_material")
);
```

28. Tabela TB_TIPO_PERSONALIZACAO

Armazena os tipos de personalização possíveis para os produtos. Exemplo: cor do forro, cores da madeira, com ou sem puxador etc.

5.2.28.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_personalizacao	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de personalização for cadastrado no sistema.
	tipo	varchar(50)	Tipos de personalizações possíveis para os produtos cadastrados. Exemplo: cor do forro, com ou sem alça, cor das madeiras etc.

5.2.28.2. Definição

```
CREATE TABLE "TB_TIPO_PERSONALIZACAO" (
```

```
  "id_tipo_personalizacao" SERIAL UNIQUE,
  "tipo" VARCHAR(50) NULL,
  PRIMARY KEY("id_tipo_personalizacao")
);
```

29. Tabela TB_TIPO_PESSOA

Armazena os possíveis tipos de usuários que podem usufruir do sistema. Exemplo: cliente, vendedor, administrador, fornecedor etc.

5.2.29.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_pessoa	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de usuário for cadastrado no sistema.
	tipo	varchar(20)	Tipos de usuários cadastrados. Exemplo: cliente, vendedor, administrador, fornecedor etc.

5.2.29.2. Definição

```
CREATE TABLE "TB_TIPO_PESSOA" (
  "id_tipo_pessoa" SERIAL UNIQUE,
  "tipo" VARCHAR(20) NULL,
  PRIMARY KEY("id_tipo_pessoa")
);
```

30. Tabela TB_TIPO_ PRODUTO

Armazena os tipos de produtos existentes. Exemplo: mesa de xadrez, caixa, bandeja, porta vinho, quadro etc.

5.2.30.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_tipo_produto	serial; unique	Esta coluna é preenchida automaticamente cada vez que um novo tipo de produto for cadastrado no sistema.
	tipo	varchar(40)	Tipos de produtos cadastrados no sistema.

			Exemplo: mesa de xadrez, bandeja, porta vinho, caixa, quadro etc.

5.2.30.2. Definição

```
CREATE TABLE "TB_TIPO_PRODUTO" (
  "id_tipo_produto" SERIAL UNIQUE,
  "tipo" VARCHAR(40) NULL,
  PRIMARY KEY("id_tipo_produto")
);
```

*31. **Tabela TB_UF***

Armazena os dados referentes às unidades federativas do Brasil.

5.2.31.1. Atributos

CHAVE	NOME	TIPO	DESCRIÇÃO
	id_uf	serial; unique	Esta coluna é preenchida automaticamente cada vez que uma nova unidade da federação for cadastrada no sistema.
	sigla	varchar(20)	Sigla da UF. Exemplo: PE, RJ, SP etc.
	nome	varchar(20)	Nome da UF. Exemplo: Pernambuco, Rio de Janeiro, São Paulo etc.

5.2.31.2. Definição

```
CREATE TABLE "TB_UF" (
  "id_uf" SERIAL UNIQUE,
  "sigla" VARCHAR(20) NULL,
  "nome" VARCHAR(20) NULL,
  PRIMARY KEY("id_uf")
);
```

6. AMBIENTE DE DESENVOLVIMENTO E IMPLEMENTAÇÃO

6.1. Apache

Um sistema que é desenvolvido para ambiente *web* necessita de um servidor *web* para hospedá-lo e para permitir sua visualização, isto é, o servidor web é o programa responsável por disponibilizar páginas, imagens, ou qualquer outro tipo de objeto ao navegador do cliente. Ele também pode operar recebendo dados do cliente, processando e enviando o resultado para que o cliente possa tomar a ação desejada (como em aplicações CGI, banco de dados web, preenchimento de formulários etc.).

O Apache foi escolhido para ser o servidor web do sistema porque ele possui suporte ao PHP, que é a linguagem em que o sistema foi programado, e já têm sua eficiência comprovada uma vez que é o servidor web mais utilizado do mundo. Segundo dados da Netcraft Web Server (http://www.netcraft.com/) mais de 47% dos servidores ativos no mundo usavam o Apache em dezembro de 2007.

O Apache é um servidor web extremamente configurável, principalmente através de seu arquivo de configuração (http.conf), robusto e de alta performance. Ele é desenvolvido por uma equipe de voluntários (conhecida como Apache Group) buscando criar um servidor web com muitas características e com código fonte disponível gratuitamente via Internet. É conveniente ressaltar que o Apache só funciona se não tiver outro servidor web ativo na máquina.

No Apache para UNIX, as instruções são enviadas através de linha de comando, entretanto para ambiente Windows, o Apache possui um aplicativo que gerencia o serviço, o Apache Service Monitor, que facilita a inicialização, finalização e re-inicialização do servidor.

A seguir, estão algumas características que fazem dele o preferido entre os administradores de sistemas:

- Possui suporte a *scripts* cgi usando linguagens como *Perl, PHP, Shell Script, ASP etc*.
- Suporta autorização de acessos, podendo ser especificadas restrições de acesso separadamente para cada endereço/arquivo/diretório acessado no servidor.
- Possibilita a autenticação requerendo um nome de usuário e senha válidos para acesso a alguma página/subdiretório/arquivo.

- Permite a personalização de logs.
- Exibe mensagens de erro.
- Suporta virtual hosting (é possível servir duas ou mais páginas com endereços/ portas diferentes através do mesmo processo ou usar mais de um processo para controlar mais de um endereço).
- Suporta IP virtual hosting.
- Suporta name virtual hosting.
- Suporta servidores Proxy ftp e http, com limite de acesso e caching (todas flexivelmente configuráveis).
- Suporta proxy e redirecionamentos baseados em URL para endereços internos.
- Suporta criptografia via SSL e certificados digitais
- Possui módulos DSO (Dynamic Shared Objects), que permitem adicionar/remover funcionalidades e recursos sem necessidade de recompilação do programa.

6.2. PHP

Uma vez que um dos requisitos do sistema era que ele fosse desenvolvido para ambiente web, era necessário utilizar uma linguagem de programação para esta finalidade. O PHP foi escolhido para implementar esse sistema por ser uma linguagem gratuita e de larga utilização para desenvolvimento de sítios, o que facilita muito a busca por referências, fóruns de usuários e tutoriais.

Para utilizar o PHP, é necessário instalar o interpretador do PHP na máquina que funcionará como servidor, e configurar o servidor web para reconhecê-lo.

PHP, que é um acrônimo para "Hypertext Preprocessor", é uma linguagem interpretada de *script*, código aberto, de uso geral, muito utilizada e especializada para o desenvolvimento de aplicações Web. Ela também pode ser caracterizada por sua portabilidade, já que seu código pode ter sido escrito em uma máquina e ser migrado para uma outra com sistema operacional diferente. Por exemplo, seu código pode ter sido escrito em uma máquina Windows e ser migrado para uma máquina Unix sem quaisquer alterações.

Como foi indicado no parágrafo anterior, o PHP pode ser utilizado na maioria dos sistemas operacionais, incluindo Linux, FreeBSD, Solaris, Microsoft Windows, Mac OS e provavelmente outros. O PHP também é suportado pela maioria dos servidores web atuais, incluindo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape,

Xitami e muitos outros. O PHP, portanto, dá a liberdade para que o programador possa escolher o sistema operacional e o servidor web.

Os programas escritos em PHP são diferentes de *scripts* CGI escritos em outras linguagens compiladas como Perl ou C. Ao invés de escrever um programa com diversos comandos para imprimir HTML, basta escrever um arquivo HTML com o código PHP inserido para realizar as instruções que se deseja. O código PHP é delimitado por tags iniciais e finais que lhe permitem pular pra dentro e pra fora do "modo PHP". Pode-se dizer que o PHP gera código HTML dinamicamente.

Usando PHP, pode-se escolher entre utilizar programação estrutural ou programação orientada a objeto, ou ainda uma mistura delas. Muitos recursos de uma programação orientada a objeto foram adicionados na versão 5 do interpretador. Além disso, o desenvolvedor não está limitado a gerar somente HTML. As habilidades do PHP incluem geração de imagens, arquivos PDF e animações Flash.

O que distingue o PHP de linguagens como Javascript, é que ele é executado no lado do servidor, enquanto que o Javascript é executado no lado do cliente. Um cliente que esteja acessando uma página escrita em PHP receberia os resultados da execução desse *script* em linguagem HTML, sem nenhum modo de determinar como é o código fonte. O servidor pode inclusive ser configurado para processar todos os seus arquivos HTML como PHP, e então não haverá nenhum modo dos usuários descobrirem se essa linguagem está sendo utilizada ou não.

Abaixo estão resumidas algumas características do PHP que indicam sua eficiência:

- É muito eficiente. Um indicador dessa eficiência é o número crescente de usuários utilizando PHP;
- É interpretado no servidor, com isso basta ao cliente possuir um navegador para poder exibir a resposta solicitada ao servidor;
- Minimiza o tráfego da rede ao limitar a necessidade de um constante diálogo entre o navegador e o servidor;
- Reduz o tempo de carregamento das páginas porque, ao final do processo, há o download de somente código HTML puro;
- Evita problemas de compatibilidade entre navegadores; qualquer navegador entende HTML. O próprio navegador do cliente é capaz de exibir a resposta emitida pelo servidor;

- Melhora as condições de segurança, pois códigos de programação nunca serão vistos pelo cliente;
- Portável, compatível com diversos sistemas operacionais: Linux, FreeBSD, Solaris, Microsoft Windows, Mac OS;
- Possui interface para diferentes sistemas de banco de dados: PostgreSQL, MySQL, mSQL, InterBase, Oracle, Firebird etc. ou qualquer outro banco de dados que suporte ODBC;
- Suportado por vários servidores web: Apache, IIS, Personal Web Server, Netscape, Xitami etc.;
- Apresenta bibliotecas integradas para muitas tarefas comuns na web, ou seja funções pré-programadas.

6.3. PostgreSQL

O PostgreSQL é um SGBD (Sistema Gerenciador de Banco de Dados) objeto-relacional de código aberto, com mais de 15 anos de desenvolvimento. É extremamente robusto e confiável, além de ser extremamente flexível e rico em recursos. Ele é considerado objeto-relacional por implementar, além das características de um SGBD relacional, algumas características de orientação a objetos, como herança e tipos personalizados. A equipe de desenvolvimento do PostgreSQL sempre teve uma grande preocupação em manter a compatibilidade com os padrões SQL92/SQL99.

1. Histórico

O PostgreSQL é um dos resultados de uma ampla evolução que se iniciou com o projeto Ingres, desenvolvido na Universidade de Berkeley, Califórnia. O lider do projeto, Michael Stonebraker, um dos pioneiros dos bancos de dados relacionais, deixou a universidade em 1982 para comercializar o Ingres, porém retornou a ela logo em seguida.

Após seu retorno a Berkeley, em 1985, Stonebraker começou um projeto pós-Ingres com o objetivo de resolver problemas com o modelo de banco de dados relacional. O principal problema era a incapacidade do modelo relacional compreender “tipos” (atualmente, chamados de objetos), ou seja, combinações de dados simples que formam uma única unidade.

O projeto resultante, chamado Postgres, era orientado a introduzir a menor quantidade possível de funcionalidades para completar o suporte a tipos. Estas funcionalidades incluiam a

habilidade de definir tipos, mas também a habilidade de descrever relações - as quais até este momento eram amplamente utilizadas mas completamente mantidas pelo usuário. No Postgres, o banco de dados "compreendia" as relações e podia obter informações de tabelas relacionadas utilizando regras.

Iniciando em 1986, a equipe divulgou uma série de documentos descrevendo a base do sistema e em 1988 o projeto possuía um protótipo funcional. A versão 1 foi liberada para um grupo pequeno de usuários em junho de 1989, seguida pela versão 2 com um sistema de regras reescrito em junho de 1990. Para a versão 3, liberada em 1991, o sistema de regras foi reescrito novamente, mas também foram adicionados suporte para múltiplos gerenciadores de armazenamento e um melhorado motor de consultas. Já em 1993, Postgres havia crescido imensamente em popularidade e possuía uma grande demanda por suporte e por novas funcionalidades. Após a liberação da versão 4, a qual era uma simples versão de limpeza, o projeto foi oficialmente abandonado pela Universidade de Berkeley.

Entretanto, devido ao fato do seu código fonte estar sob uma licença BSD, o seu desenvolvimento foi continuado. Em 1994, dois estudantes de Berkeley, Andrew Yu e Jolly Chen, adicionaram um interpretador SQL para substituir a linguagem QUEL (desenvolvida para o Ingres) e o projeto foi renomeado para Postgres95. Com a divulgação de seu código pela Internet, Postgres95 iniciou uma nova vida como *software open source*.

Em agosto de 1996, Marc Fournier, Bruce Momjian e Vadim B. Mikheev lançaram a primeira versão externa da Universidade de Berkeley e deram início à tarefa de estabilizar o código herdado. Também em 1996, o projeto foi renomeado para PostgreSQL a fim de refletir a nova linguagem de consulta ao banco de dados: SQL. A primeira versão de PostgreSQL, a 6.0, foi liberada em janeiro de 1997. Desde então, um grupo de desenvolvedores e de voluntários de todo o mundo, coordenados pela Internet, têm mantido o *software* e desenvolvido novas funcionalidades.

As principais características acrescentadas nas versões 6.x são o en:MVCC (Multiversion Concurrency Control – Controle de Concorrência Multiversões), melhorias no SQL e novos tipos de dados nativos (novos tipos de datas e hora e tipos geométricos).

Em maio de 2000 foi liberada a versão 7.0. As versões 7.x trouxeram as seguintes novas funcionalidades: Write-Ahead Log (WAL), esquemas SQL, outer joins, suporte a IPv6, indexação por texto, suporte melhorado a SSL e informações estatísticas do banco de dados.

A versão 8.0 foi lançada em janeiro de 2005 e entre outras novidades, foi a primeira a ter suporte nativo para Microsoft Windows (tradicionalmente, o PostgreSQL só rodava de

forma nativa em sistemas Unix e, em sistemas Windows - através da biblioteca Cygwin). Dentre as muitas novidades da versão 8.x, pode-se destacar o suporte a tablespaces, savepoints, point-in-time recovery, roles e Two-Phase Commit (2PC). Em fevereiro de 2008 foi lançada a versão mais recente: 8.3.

2. O PostgreSQL hoje

A equipe do projeto cresceu e se espalhou pelo mundo. O Grupo Global de Desenvolvimento do PostgreSQL tem membros nos Estados Unidos, Canadá, Japão, Rússia, vários países da Europa e alguns outros. Esse grupo é formado essencialmente por empresas especializadas em PostgreSQL, empresas usuárias do sistema, além dos pesquisadores acadêmicos e programadores independentes. Além da programação, essa comunidade é responsável pela documentação, tradução, criação de ferramentas de modelagem e gerenciamento, e elaboração de extensões e acessórios.

Pela riqueza de recursos e conformidade com os padrões, ele é um SGBD muito adequado para o estudo universitário do modelo relacional, além de ser uma ótima opção para empresas implementarem soluções de alta confiabilidade sem altos custos de licenciamento. É um programa distribuído sob a licença BSD, o que torna o seu código fonte disponível e o seu uso livre para aplicações comerciais ou não. O PostgreSQL foi implementado em diversos ambientes de produção no mundo, entre eles, um bom exemplo do seu potencial é o banco de dados que armazena os registros de domínio .org, mantido pela empresa Afilias.

3. Alguns Recursos

- Sub-consultas;
- Controle de concorrência multi-versão (MVCC);
- Integridade Referencial;
- Funções armazenadas (*Stored Procedures*), que podem ser escritas em várias linguagens de programação (PL/PgSQL, Perl, Python, Ruby, e outras);
- Gatilhos (*Triggers*);
- Tipos definidos pelo usuário;
- Esquemas (*Schemas*);
- Conexões SSL;

- Áreas de armazenamento (*Tablespaces*);
- Pontos de salvamento (*Savepoints*);
- *Commit* em duas fases;
- Arquivamento e restauração do banco a partir de logs de transação;
- Diversas ferramentas de replicação;
- Extensões para dados geoespaciais, indexação de textos, xml e várias outras.

6.4. Dreamweaver MX 2004

A linguagem HTML é usada para escrever páginas de documentos para Web ou WWW. Com essa linguagem, que para alem do texto, tem comandos para introdução de imagens, formulários, alteração de fontes etc, podem-se definir páginas que contenham informação nos mais variados formatos: texto, som, imagens e animações.

O Dreamweaver foi o editor de HTML utilizado para programar o sistema. Ele possui três ambientes de desenvolvimento: visual, texto e misto. A interface visual foi utilizada para configurarmos os layouts das páginas. Enquanto que a interface de texto foi utilizada para escrever os códigos em linguagem PHP que são inseridos no código HTML e escrever os arquivos com as funções em Javascript.

O Javascript é uma linguagem para páginas web, desenvolvida pela Netscape. É executada do lado do cliente, porque é o navegador que suporta a carga de processamento. Com essa linguagem, é possível validar formulários e adicionar recursos dinâmicos às páginas HTML, ou seja, é possível ao usuário interagir com a página, seja respondendo enquetes, seja respondendo perguntas eletrônicas, seja vendo conteúdo randômico, dentre outras formas. Em sua essência, a linguagem Javascript atua inserida no meio do código HTML de páginas *Web*. Essa inserção pode ser feita de várias formas, desde a inclusão de código numa área determinada ou a inserção em vários pontos da página. Tudo depende da utilidade necessária. O fato do Javascript não exigir a instalação de *softwares* especiais para execução (assim como ocorre com linguagens mais sofisticadas, como PHP e ASP) o fez ter grande popularidade, já que qualquer página HTML pode conter recursos em Javascript. Tudo depende, basicamente, da capacidade do navegador em entender as instruções.

6.5. Jude / Community

O Jude / Community foi o *software* usado para desenvolver a modelagem da análise e do projeto do sistema. É uma ferramenta de modelagem visual, baseada na notação da UML

(*Unified Modeling Language*), que permite definir a arquitetura do *software* através dos diagramas de Casos de Uso, de classe e de objeto; o seu comportamento através dos diagramas de seqüência, colaboração, atividade, estado, componentes e implantação.

Há outras ferramentas com essas funcionalidades, mas o Jude foi usado por ser gratuito e ter certa credibilidade nesse ramo. No decorrer do processo, foram encontrados alguns erros no Jude, mas nada muito grave e que pudesse comprometer a modelagem do sistema.

7. CONSIDERAÇÕES FINAIS

7.1. Conclusão

Esse trabalho apresentou a análise, a modelagem e o desenvolvimento da primeira versão do sítio www.marchetaweb.com.br, que pretende disponibilizar artesanato personalizado com madeira na rede.

O sistema foi desenvolvido em *web* por ser um *e-commerce* e precisar desse meio para interagir com seus eventuais clientes.

Apesar da utilização da lógica de programação estruturada foram criadas classes que fazem a comunicação com o banco de dados do sistema. Por exemplo, há uma classe que faz exclusivamente a conexão com o banco, isolando-a do resto da aplicação que está voltada às regras do negócio. Apesar de ter sido utilizada lógica estruturada, a documentação e a modelagem do sistema foram feitas através da análise orientada a objeto, o que torna mais clara a compreensão do sistema e tem sido cada vez mais utilizada na modelagem dos novos sistemas. Um sistema composto por objetos facilita o trabalho em equipe porque pode-se definir quais classes que cada pessoa desenvolverá e, através dos diagramas do modelo, elas poderão entender como as classes se comunicam, ou seja, não há necessidade de se conhecer o funcionamento interno de cada classe, elas terão uma visão mais abrangente do sistema.

A modelagem foi feita através da UML, que é uma linguagem de diagramação ou notação padronizada para especificar, visualizar e documentar modelos de sistemas de *software* orientados a objeto. Ela possui uma notação visual que facilita o trabalho e a discussão com o cliente porque permite a compreensão do sistema em um nível mais elevado de abstração que independe de como ela será implantada. Nesse sentido, tem-se como objetivo usá-la com mais propriedade e projetar classes que possam ser reaproveitadas em outros projetos para ganhar tempo e minimizar esforço.

A arquitetura, dentro dos limites das ferramentas utilizadas, foi dividida em três camadas: Apresentação, Controle e Dados. Deste modo, caso sejam encontrados erros no sistema, a solução dos mesmos dar-se-á mais rapidamente, sem que haja a necessidade de modificação de muitas partes da aplicação. Durante o desenvolvimento, sempre houve a preocupação em projetar um sistema versátil e manutenível, por isso que se procurou adotar soluções que permitissem a inclusão de novos módulos sem causar impacto. Essa preocupação também se estendeu à modelagem do banco de dados, já que foi o mesmo foi

projetado para poder suportar a inclusão de novas entidades sem haver comprometimento da sua integridade.

A opção por utilizar *softwares* livres tem funcionado. São ferramentas gratuitas e eficientes que sempre estão evoluindo, possuem uma documentação completa e disponibilizada em seus sites oficiais e com elas tem-se conseguido obter soluções que atendem às necessidades do cliente.

Pode-se comprovar também que a portabilidade entre o PHP e o PostgreSQL está bem consolidada. O *software* pôde ser migrado tranqüilamente de um sistema operacional Windows para um Unix, usado no servidor onde o sítio encontra-se atualmente hospedado. Isso já era esperado, uma vez que o *software* é dirigido para navegadores, bem como também poderia ter sido migrado para outras distribuições baseadas em FreeBSD, Linux, Solaris ou Mac OS, desde que elas tenham suporte a PHP e PostgreSQL.

7.2. Trabalhos Futuros

Como ficou explicitado na conclusão desse trabalho, o sistema foi desenvolvido em uma versão inicial, portanto existirão alterações constantes no sistema. Essas futuras alterações se aplicarão às três camadas do sistema, principalmente a de apresentação, que nessa fase inicial não foi muito realçada, devido ao fato de se tratar de um projeto inicialmente com ênfase de trabalho universitário de conclusão de curso.

Por se tratar de um sistema dinâmico, as alterações sempre serão necessárias. Contudo, devido ao fato dele ser altamente baseado em orientação a objetos e dividido em camadas, as alterações não serão muito custosas.

Uma primeira alteração no sistema seria a inclusão de mais figuras dos modelos de produtos, pois assim o cliente tem mais possibilidade de conhecer o que está sendo encomendado.

Na camada de dados, haverá a necessidade de se pensar em algumas alterações para que se evite uma sobrecarga de dados inúteis ao sistema. Uma solução que já está sendo estudada é a retirada dos cadastros de itens de pedidos que não tenham se transformado efetivamente em pedidos. Para isso, deve-se verificar com o qual seria o prazo temporal no qual esses itens possam ser considerados desnecessários.

Algumas validações também deverão ser incluídas no sistema, como por exemplo, confirmações nos momentos de exclusões de certos cadastros, dentre outras.

8. REFERÊNCIAS

PRESSMAN, Roger S. **Engenharia de software**. 5. ed. Rio de Janeiro: McGraw-Hill, 2002.

GUEDES, Gilleanes T. A. **UML**: uma abordagem prática. 2. ed. São Paulo: Novatec Editora, 2006.

BOOCH, Grady **UML**: guia do usuário. 2. ed. Rio de Janeiro: Campus, 2005.

APACHE HTTP Server: servidor versão 2.2.8. The Apache Software Foundation. Disponível em: <http://httpd.apache.org/download.cgi>. Acesso em: 25 janeiro 2008.

PHP: linguagem de *script* versão 5.2.5. The PHP Group, 2001-2008. Disponível em: <http://www.php.net/downloads.php>. Acesso em: 25 janeiro 2008.

POSTGRESQL: banco de dados versão 8.2.4.1. The PostgreSQL Global Development Group, 1996-2005. Disponível em: <http://www.postgresql.org/download/>. Acesso em: 13 agosto 2007.

MICROSOFT OFFICE WORD 2007: editor de texto versão 12.0.6212.1000 SP 1 MSO 12.0.6213.1000. Microsoft Corporation, 2006. Disponível em: <http://msdn2.microsoft.com/pt-br/subscriptions/default.aspx>. Acesso em: 16 fevereiro 2008.

JUDE Community: editor de UML versão 5.1.1 (Model Version: 26). Change Vision, Inc, 2006-2007. Disponível em: <https://jude.change-vision.com/jude-web/product/community.html>. Acesso em: 25 janeiro 2008.

DB Designer 4: editor visual para criação de banco de dados versão 4.0.5.6. fabFORCE.net Copyright, 2003. Disponível em: <http://www.fabforce.net/dbdesigner4>. Acesso em: 14 agosto 2007.

W3Schools - JavaScript Tutorial: tutorial online de Javascript. Disponível em: <http://www.w3schools.com/js/default.asp>. Acesso em: 21 outubro 2007.

Wikipédia, a enciclopédia livre – Enciclopédia livre na internet. Disponível em: <http://pt.wikipedia.org/wiki/Marchetaria>. Acesso em: 21 junho 2008.

Antiqua Marchetaria - Sítio sobre marchetaria. Disponível em: <http://www.antiquamarchetaria.com/Historia.htm>. Acesso em: 21 junho 2008.

Wikipédia, a enciclopédia livre – Enciclopédia livre na internet. Disponível em: <http://pt.wikipedia.org/wiki/Servidor_Apache>. Acesso em: 24 junho 2008.

Wikipédia, a enciclopédia livre – Enciclopédia livre na internet. Disponível em: <http://pt.wikipedia.org/wiki/PostgreSQL>. Acesso em: 24 junho 2008.

PostgreSQL Brasil. Disponível em: <http://www.postgresql.org.br/Introdução_e_histórico>. Acesso em: 24 junho 2008.

ÍNDICE REMISSIVO